EMC²

Building Secure SANs

Version 4.0

- Building Secure SANs
- Mechanisms to Enhance SAN Security
- Implementing Product-Specific Security Mechanisms
- Implementing Fabric-Based Encryption

Ron Dharma
Veena Venugopal
Sowjanya Sake
Vin Dinh

Copyright © 2011 - 2013 EMC Corporation. All rights reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

EMC^2, EMC, and the EMC logo are registered trademarks or trademarks of EMC Corporation in the United State and other countries. All other trademarks used herein are the property of their respective owners.

For the most up-to-date regulator document for your product line, go to EMC Online Support (https://support.emc.com).

Part number H8082.3

Contents

Preface 9

Chapter 1 Building Secure SANs

Understanding how to secure your SANs 16
- Basic security concepts 17
- Interconnecting storage 18

Security attacks against SANs 20
- Snooping (Eavesdropping) 20
- Spoofing (modification/fabrication) 22
- Denial-of-service (Disruption) 22

Secure SAN architecture, components, and mechanisms 24
- SAN architectures 25
- Security components 26
- Security mechanisms 26

Building secure SANs 52
- Design considerations 52

Chapter 2 Implementing RSA Key Manager (RKM) HA Functionality

RSA Key Manager (RKM) overview 56
Configuring the monitor 58

Chapter 3 Security Implementation Examples

EMC Celerra iSCSI data security 66
- Supported features 66
- Best practices 66
- References 68

Brocade 69
Security features 69
EMC conditionally or unsupported features 71
Best practices 72
Related Brocade documentation 74
Brocade M Series 76
Security features 77
Best practices 79
Cisco 82
Security features 82
Conditionally or unsupported features 83
Best practices 84
Related Cisco documentation 86

Chapter 4 Cisco MDS 9000 Family Storage Media Encryption (SME)

Overview 90
Key features 91
Supported key features 92
Terminology 93
Topology guidelines 94
Requirements 94
General guidelines 95
Sizing guidelines 95
Configuration limits 96
Prerequisites for configuring SME 96
Core-Edge topology 97
Single fabric SME cluster deployment 99
Zoning requirements 102
FC-Redirect requirements 102
Configuration requirements 102
Key hierarchy overview 103
Key types 103
Levels of security 105
Key managers 105
Key management best practice 108
Cisco SME tape configuration 108
Implementation best practices 110

Chapter 5 Cisco SME Configuration in a Cisco Key Manager Environment

Overview 112
SAN 113
Key management 114
- Key Manager 114
- Master key security selection 116
- Tape media specific key settings 117
- Tape recycling 118

IP network 119
- Securing the management of switches to the FMS 119
- Securing the web client communications to the FMS 120
- Securing the MDS to KMC communication through SSL .. 121

Chapter 6 Cisco Key Management Center (KMC) Migration Procedure

Issue and solution overview 124
- Issue 124
- Solution 124

Step 1: Migration from two unique KMCs 127
Step 2: Periodic backup and restore of the database 137
Step 3: Disaster recovery procedure 138
- Case 1: KMC-A failure 138
- Case 2: Complete site failure, KMC-A and SME-A cluster 143

Chapter 7 Brocade Encryption Switch/Blade

Introduction 152
Fabric-based encryption solution terms and concepts 153
Encryption topology basics 160
- Estimating the number of Encryption Engines needed 160
- Determining the placement of Encryption Engines 161
- Firmware level 161
- LAN assessment 162
- RSA Key Manager (RKM) key vaults 163

Brocade fabric-based encryption case study 164
- Important notes 164
- Target topology 165
- Summary of installation steps 165
- Summary of initialization steps 168
- Summary of CTCs, hosts and LUNs 220

TimeFinder case study 221
Configuration requirements 221
Reference architecture 222
Summary of initialization steps 223
Rekeying encrypted source LUNs in the TimeFinder environment 231
SRDF encryption case study 232
Configuration requirements 233
Reference architecture 234
Summary of initialization steps 235
Configuring the Encryption Engines 236
Configuring the Encryption Groups 242
Configuring the High Availability (HA) clusters 246
Setting up RKM key vault 248
Setting up ADX IPLB (IP Load Balance) 256
Registering the RKM KV on the Encryption Group leader 262
Dealing with certificate expiration (KAC, Apache Server, and CA) 266
Generating and backing up the EG Master key 272
Ensuring that both fabrics are set for defzone--noaccess 276
Enabling remote replication mode 277
Creating the CTCs at each site 278
Adding the source SRDF LUNs to each CTC at Site 1 280
Adding the source SRDF LUNs to each CTC at Site 2 284
RecoverPoint encryption case study 293
Configuration requirements 293
Reference architecture 294
Summary of initialization steps 295
Rekeying in the RecoverPoint environment 299

Chapter 8 Best Practices for EMC Disk Library and Encryption Switches

Overview 302
Challenges 304
Best practices for Cisco SME and Brocade Encryption Switch 305
Cisco SME 305
Brocade Encryption Switch 307
Turning compression on and off on the EDL 309

Figures

	Title	Page
1	SAN security bridges	19
2	Snooping in a SAN	21
3	Spoofing in a SAN	22
4	Denial-of-service attack	23
5	Security zones	25
6	Encryption process	34
7	Symmetric encryption example	35
8	Asymmetric encryption example	36
9	Encryption levels	41
10	Deployment of the RKM cluster with F5 BIG-IP LTM topology example	57
11	Working health monitor example	63
12	LUN masking	67
13	CHAP authentication	68
14	Cisco SME example	91
15	Core-edge topology example	97
16	Supported single fabric deployment with RKM, NX-OS 4.2.3 and earlier	100
17	Supported single fabric deployment with KMC, SAN-OS and NX-OS	101
18	KMC integrated with Cisco Fabric Manager	106
19	KMC decoupled from Cisco Fabric Manager	107
20	Storing the keys using RKM example	108
21	Brocade Encryption Switch (left) and PB-DCX-16EB encryption blade (right)	155
22	Encryption nodes and EE	155
23	Frame redirection through the encryption blade	156
24	HA clusters in a DEK cluster	157
25	LAN communication between Fabric A and Fabric B	163

26 Dual-fabric, core-edge Storage Area Network (SAN), Target topology 165
27 Initnode command creates certificates for node to be shared or exported 173
28 KAC signing and storage process 175
29 Signed KAC certificate storage process 176
30 RKM Certificate transfer to Group Leader node process 176
31 Master Key can be exported to different types of media 182
32 Final configuration of CTCs, hosts, and LUNs 220
33 TimeFinder case study architecture 222
34 Encryption for two sites with SRDF 234
35 Single subnet deployment 259
36 Routed subnet topology 260
37 Remote Server Subnet topology 262
38 RecoverPoint case study architecture 294
39 Encryption process 303
40 Cisco SME encryption example 306
41 Brocade Encryption Switch encryption example 307

Preface

This EMC Engineering TechBook identifies and exemplifies some common SAN security attacks, presents some built-in and bolt-on mechanisms to enhance SAN security, and provides some insight on how to implement various product-specific security mechanisms.

E-Lab would like to thank all the contributors to this document, including EMC engineers, EMC field personnel, and partners. Your contributions are invaluable.

As part of an effort to improve and enhance the performance and capabilities of its product lines, EMC periodically releases revisions of its hardware and software. Therefore, some functions described in this document may not be supported by all versions of the software or hardware currently in use. For the most up-to-date information on product features, refer to your product release notes. If a product does not function properly or does not function as described in this document, please contact your EMC representative.

Audience

This TechBook is intended for EMC field personnel, including technology consultants, and for the storage architect, administrator, and operator involved in acquiring, managing, operating, or designing a networked storage environment that contains EMC and host devices.

EMC Support Matrix and E-Lab Interoperability Navigator

For the most up-to-date information, always consult the *EMC Support Matrix* (ESM), available through E-Lab Interoperability Navigator (ELN) at http://elabnavigator.EMC.com, under the **PDFs and Guides** tab.

Under the **PDFs and Guides** tab resides a collection of printable resources for reference or download. All of the matrices, including the ESM (which does not include most software), are subsets of the

E-Lab Interoperability Navigator database. Included under this tab are:

- The *EMC Support Matrix*, a complete guide to interoperable, and supportable, configurations.
- Subset matrices for specific storage families, server families, operating systems or software products.
- Host connectivity guides for complete, authoritative information on how to configure hosts effectively for various storage environments.

Under the **PDFs and Guides** tab, consult the *Internet Protocol* pdf under the "Miscellaneous" heading for EMC's policies and requirements for the *EMC Support Matrix*.

Related documentation

Related documents include:

- The following documents, including this one, are available through the E-Lab Interoperability Navigator, **Topology Resource Center** tab, at http://elabnavigator.EMC.com.

  These documents are also available at the following location:

`http://www.emc.com/products/interoperability/topology-resource-center.htm`

  - *Backup and Recovery in a SAN TechBook*
  - *Extended Distance Technologies TechBook*
  - *Fibre Channel over Ethernet (FCoE): Data Center Bridging (DCB) Concepts and Protocols TechBook*
  - *Fibre Channel SAN Topologies TechBook*
  - *iSCSI SAN Topologies TechBook*
  - *Networked Storage Concepts and Protocols TechBook*
  - *Networking for Storage Virtualization and RecoverPoint TechBook*
  - *WAN Optimization Controller Technologies TechBook*
  - *EMC Connectrix SAN Products Data Reference Manual*
  - *Legacy SAN Technologies Reference Manual*
  - *Non-EMC SAN Products Data Reference Manual*
- *EMC Support Matrix*, available through E-Lab Interoperability Navigator at http://elabnavigator.EMC.com > **PDFs and Guides**
- RSA security solutions documentation, which can be found at http://RSA.com > **Content Library**

All of the following documentation and release notes can be found at EMC Online Support (https://support.emc.com). From the toolbar, select **Support > Technical Documentation and Advisories**, then choose the appropriate Hardware/Platforms, Software, or Host Connectivity/HBAs documentation links.

EMC hardware documents and release notes include those on:

- Connectrix B series
- Connectrix MDS (release notes only)
- VNX series
- CLARiiON
- Celerra
- Symmetrix

EMC software documents include those on:

- ControlCenter
- RecoverPoint
- TimeFinder
- PowerPath

The following E-Lab documentation is also available:

- Host Connectivity Guides
- HBA Guides

For Cisco and Brocade documentation, refer to the vendor's website.

- http://cisco.com
- http://brocade.com

Authors of this TechBook

This TechBook was authored by Ron Dharma, Veena Venugopal, Sowjanya Sake, and Vinh Dinh with contributions from EMC engineers, EMC field personnel, and partners.

Ron Dharma is a Principal Integration Engineer and team lead for the Advance Product Solution group in E-Lab. Prior to joining EMC, Ron was a SCSI software engineer, spending almost 11 years resolving integration issues in multiple SAN components. He dabbled in almost every aspect of the SAN including storage virtualization, backup and recovery, point-in-time recovery, and distance extension. Ron provided the original information in this document, and works with other contributors to update and expand the content.

Veena Venugopal is a Senior Systems Integrations Engineer and has been at EMC for over 7 years. Veena works with new technologies as they enter the E-Lab, maturing them before extending support to customers. She is responsible for encryption projects, third-party clustered filesystems, and third-party storage virtualization.

Sowjanya Sake is a a Senior Systems Integration engineer with over 6 years of experience in storage technologies, tape virtualization, backup and recovery, high availability, and tape and disk libraries. Currently, Sowji qualifies the tape and disk library with Celerra ndmp backup, including EMC disk library, Quantum Dxi, data domain VTLs, Scalar i2000, i6k, i40/80, i500, and StorageTek tape libraries, in combination with various Brocade and Cisco Switches, for EMC's E-Lab. Previously, Sowji worked on Storagetek Virtual Storage Manager and Brocade Fibre Channel switches.

Vinh Dinh is a Principal Integration Engineer and has been with EMC for over 16 years. For the past several years, Vinh has worked in the E-Lab evaluating new storage and networking technologies. Vinh's work encompasses FC, iSCSI, FCoE, and Infiniband. Vinh is also involved in evaluating and qualifying various data encryption products.

Conventions used in this document

EMC uses the following conventions for special notices:

IMPORTANT

An important notice contains information essential to software or hardware operation.

Note: A note presents information that is important, but not hazard-related.

Typographical conventions

EMC uses the following type style conventions in this document.

Normal

Used in running (nonprocedural) text for:

- Names of interface elements (such as names of windows, dialog boxes, buttons, fields, and menus)
- Names of resources, attributes, pools, Boolean expressions, buttons, DQL statements, keywords, clauses, environment variables, functions, utilities
- URLs, pathnames, filenames, directory names, computer names, filenames, links, groups, service keys, file systems, notifications

Bold	Used in running (nonprocedural) text for: • Names of commands, daemons, options, programs, processes, services, applications, utilities, kernels, notifications, system calls, man pages Used in procedures for: • Names of interface elements (such as names of windows, dialog boxes, buttons, fields, and menus) • What user specifically selects, clicks, presses, or types
Italic	Used in all text (including procedures) for: • Full titles of publications referenced in text • Emphasis (for example a new term) • Variables
`Courier`	Used for: • System output, such as an error message or script • URLs, complete paths, filenames, prompts, and syntax when shown outside of running text
`Courier bold`	Used for: • Specific user input (such as commands)
`Courier italic`	Used in procedures for: • Variables on command line • User input variables
< >	Angle brackets enclose parameter or variable values supplied by the user
[ ]	Square brackets enclose optional values
|	Vertical bar indicates alternate selections - the bar means “or”
{ }	Braces indicate content that you must specify (that is, x or y or z)
...	Ellipses indicate nonessential information omitted from the example

Where to get help

EMC support, product, and licensing information can be obtained on the EMC Online Support site, as described next.

Note: To open a service request through the EMC Online Support site, you must have a valid support agreement. Contact your EMC sales representative for details about obtaining a valid support agreement or to answer any questions about your account.

Product information

For documentation, release notes, software updates, or for information about EMC products, licensing, and service, go to the EMC Online Support site (registration required) at:

https://support.EMC.com

Technical support

EMC offers a variety of support options.

Support by Product — EMC offers consolidated, product-specific information on the Web at:

https://support.EMC.com/products

The Support by Product web pages offer quick links to Documentation, White Papers, Advisories (such as frequently used Knowledgebase articles), and Downloads, as well as more dynamic content, such as presentations, discussion, relevant Customer Support Forum entries, and a link to EMC Live Chat.

EMC Live Chat — Open a Chat or instant message session with an EMC Support Engineer.

eLicensing support

To activate your entitlements and obtain your Symmetrix license files, visit the Service Center on https://support.EMC.com, as directed on your License Authorization Code (LAC) letter e-mailed to you.

For help with missing or incorrect entitlements after activation (that is, expected functionality remains unavailable because it is not licensed), contact your EMC Account Representative or Authorized Reseller.

For help with any errors applying license files through Solutions Enabler, contact the EMC Customer Support Center.

If you are missing a LAC letter, or require further instructions on activating your licenses through the Online Support site, contact EMC's worldwide Licensing team at licensing@emc.com or call:

- North America, Latin America, APJK, Australia, New Zealand: SVC4EMC (800-782-4362) and follow the voice prompts.
- EMEA: +353 (0) 21 4879862 and follow the voice prompts.

We'd like to hear from you!

Your suggestions will help us continue to improve the accuracy, organization, and overall quality of the user publications. Send your opinions of this document to:

techpubcomments@emc.com

Your feedback on our TechBooks is important to us! We want our books to be as helpful and relevant as possible. Send us your comments, opinions, and thoughts on this or any other TechBook to:

TechBooks@emc.com

1

Building Secure SANs

This chapter provides the following information to better enable you to secure your SAN:

- Understanding how to secure your SANs 16
- Security attacks against SANs 20
- Secure SAN architecture, components, and mechanisms 24
- Building secure SANs 52

Understanding how to secure your SANs

Note: This chapter discusses security in a data storage environment. Some readers may associate the acronym SANS with System Administration Networking and Security, while others are more familiar with SANs being defined as storage area networks. This chapter will use the latter definition of storage area networks whenever the acronyms SAN or SANs are used.

Security in IT has always played a significant role in the successful operation of a datacenter. An organization's security policy will often define at a high level how information should be protected from improper access or modification. These policies are traditionally implemented in various forms of network perimeter controls like ACLs, firewalls, IPS, etc., but are seldom regarded as a requirement for storage systems.

Valuable information such as intellectual property, personal identities, and financial transactions is routinely processed through, and stored in, a storage area network (SAN). The security of a SAN is critical because of the value of the extractable information. A disturbing fact that it is becoming common place is letting hosts that serve the internal and external networks also make use of the SAN. This fact changes the paradigm that hosts in the internal networks are safe since they are behind the firewalls. With the proliferation of SAN, this is no longer true and, if exploited, a vulnerable host with SAN connectivity can lead to a security disaster. Once valuable data is compromised, the physical computers and data networks become meaningless.

Designing, testing, and securing SANs is IT intensive, but necessary, to manage and protect vital information. Without proactive security measures, outsiders can easily obtain information from your SAN.

To secure your SAN you should:

- Assess configurations
- Test secure mechanism effectiveness
- Identify holes
- Quantify risks
- Implement practical security solutions for high risk exposures.

Basic security concepts

SAN configurations using block data include both Fibre Channel (FC) and Internet SCSI (iSCSI) protocols. Basic security *concepts* such as authentication, authorization, administration, and encryption (each explained briefly in this section) are similar for these protocols but *mechanisms* to protect Fibre Channel and iSCSI SANs may differ. Refer to "Secure SAN architecture, components, and mechanisms" on page 24 for more information on components and mechanisms.

Availability

Availability is the process of making data accessible in a secured manner. Availability allows data that resides in a SAN to be available only to authorized end-users, applications, servers, or network devices when requested.

Authentication

Authentication is the process of validating the identity of an entity. The authentication process normally involves a supplicant's presentation of a known credential together with an identifying element that is either known, possessed, or part of. The strength of the authentication depends on the number of factors challenged from the above-mentioned list. Authentication in a SAN is challenged on multiple fronts including switch-switch, host-switch, target-switch, and switch-storage administration.

Authorization

Authorization is the process of granting access rights and privileges to an entity that is considered trusted, usually after authentication is successful. Authorization methods in iSCSI/Fibre Channel SANs apply to hardware which is the WWN and does not allow changeable usernames. Furthermore, no secondary checks are made as this would be a weakness that could be exploited through spoofing. (See "Spoofing (modification/fabrication)" on page 22.)

Auditing

Auditing is the process of capturing and retaining events for current and future analysis. This ability to capture and retain all events about the infrastructure is essential for security awareness and overall stability. SAN uses SNMP to trap events and Storage Management Initiative Specification (SMI-S) to track and manage storage.

Integrity

Integrity is the process of ensuring that an entity can be trusted whereby it is of the exact form that was intended. For a SAN, integrity means the preservation of data that is not corrupted by intentional or unintentional means.

Encryption

Encryption is the ability to obfuscate an entity, usually data. Encryption is used as a tool to hide information from unauthorized presentation thereby providing confidentiality. In a SAN, encryption can be used in two scenarios:

- Encryption while transmitted across the wire (in-transit);
- Encryption within the storage disks (at-rest).

Interconnecting storage

The practice of interconnecting storage in the SAN can bridge several internet (TCP/IP) and security (such as DMZ, departmental, and corporate) zones that are purposefully segmented through host network access. Figure 1 on page 19 shows how a SAN can bridge external and internal security segments.

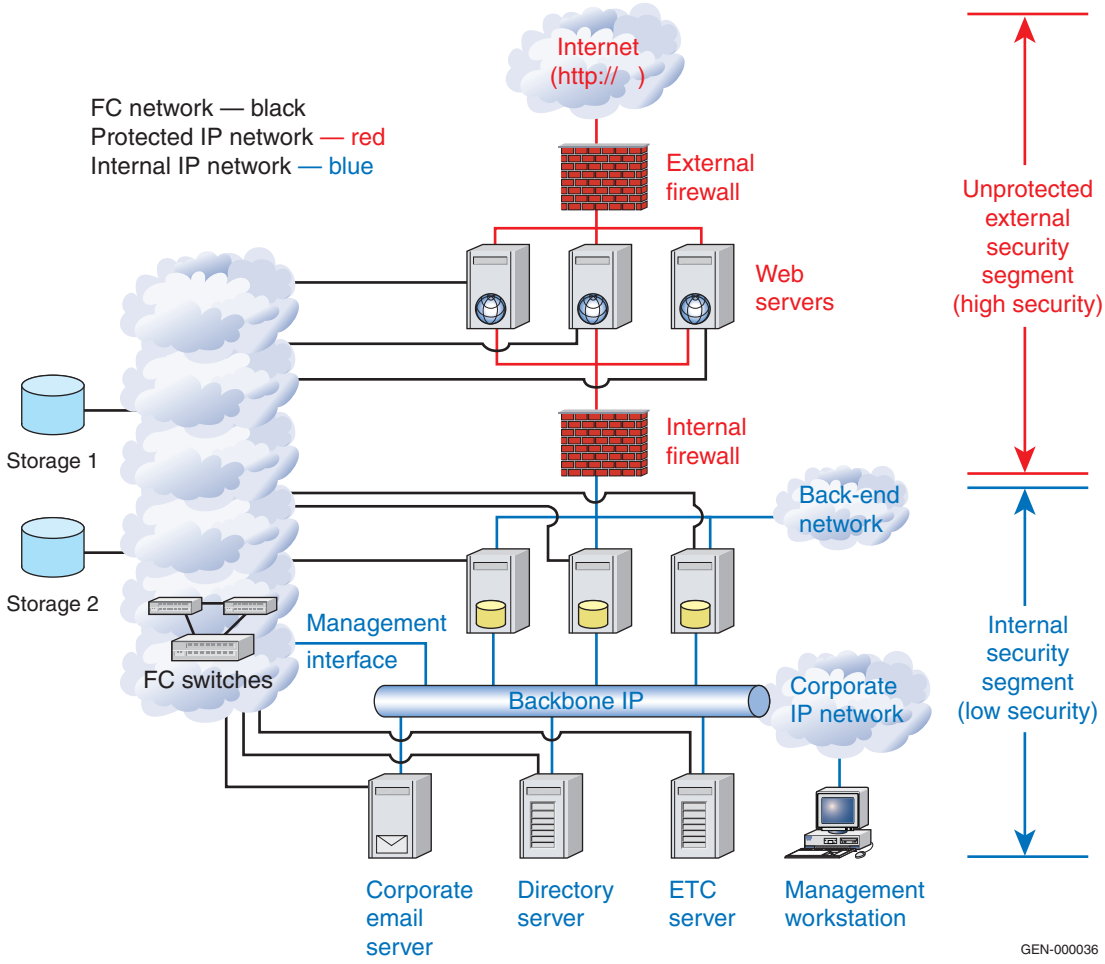


Figure 1 **SAN security bridges**

Security attacks against SANs

Security attacks against SANs are similar to security attacks against IP networks. Breaches of security can include breaches of authorization, authentication, data confidentiality, and/or data integrity. iSCSI SANs and Fibre Channel SANs have similar security flaws, including significant weaknesses with authentication and authorization.

Examples provided in this section describe some common security attacks in a SAN. These include:

- "Snooping (Eavesdropping)" on page 20
- "Spoofing (modification/fabrication)" on page 22
- "Denial-of-service (Disruption)" on page 22

Several other attacks such as Fibre Channel Name Server/iSNS pollution, session hijacking, zone hopping, E_Port replication, F_Port replication, LUN mask subversion, iSCSI Qualifier Name (IQN) spoofing, CHAP username sniffing, CHAP hash spoofing, and SAN management attacks can also expose a tremendous amount of information. These attacks can either provide the attacker with needed information for a subsequent attack, or directly compromise data. In many cases a breach of security involves a combination of security attacks.

For information on components and mechanisms to prevent security attacks, refer to "Secure SAN architecture, components, and mechanisms" on page 24.

Snooping (Eavesdropping)

Snooping is a deliberate act to access data without authorization. Different methods include, but are not limited to, eavesdropping, sniffing, session hacking, intercepting, copying, and monitoring.

Figure 2 on page 21 shows an example of snooping in a Fibre Channel SAN. Similar snooping methods can be accomplished in an iSCSI SAN. In this example, the name server table is modified by an unauthorized management session. This attack requires knowledge of the fabric switch password, which can be sniffed on the internal network if SNMPv1 or telnet was used for administration by the storage administrator. The rogue Node C will have its Source ID (S_ID) and World Wide Names (WWN) strategically placed in the

routing table and zoning table. This type of attack can be prevented with the implementation of security mechanisms to prevent unauthenticated and unauthorized manipulation of the Fibre Channel name server or iSNS. (Refer to "Secure SAN architecture, components, and mechanisms" on page 24.)

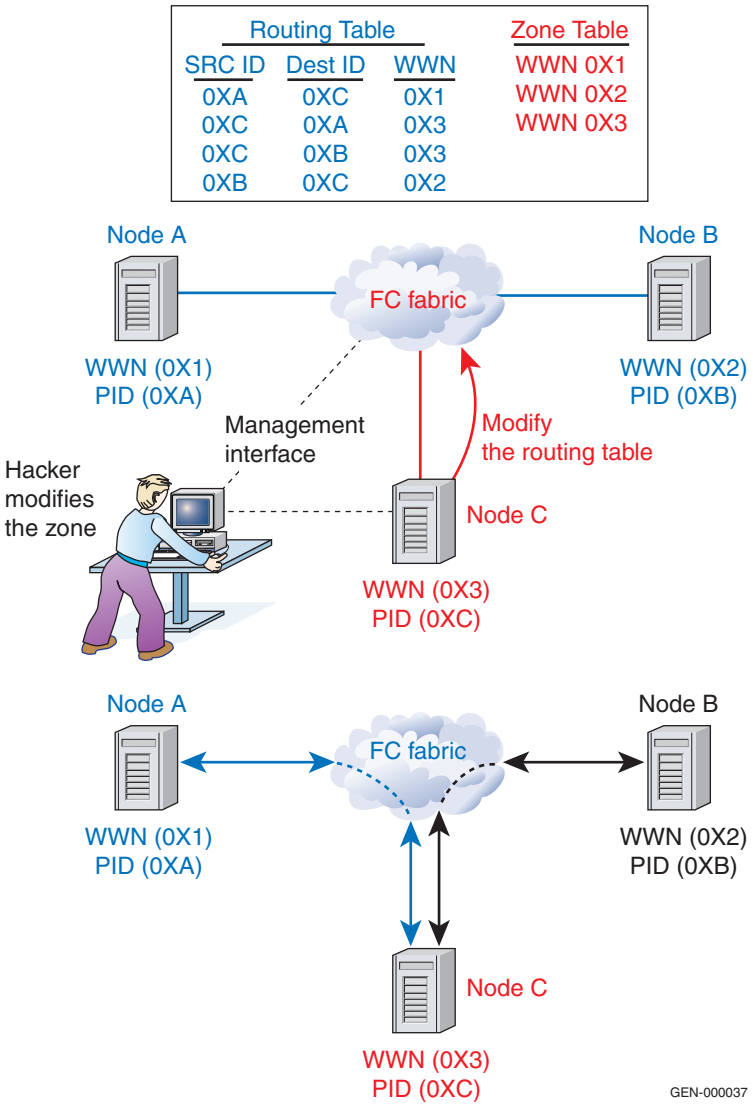


Figure 2 **Snooping in a SAN**

Spoofing (modification/fabrication)

Spoofing is a deliberate act to assume an identity in order to gain unauthorized access to the data of the company or another user. WWNs are used to identify nodes in a Fibre Channel SAN, whereas in an iSCSI SAN a node is identified by an iSCSI Qualified Name (IQN). Without proper security mechanisms in place, both are easy to change and spoof. Some methods modify part of the information on the fly or use native host bus adapter (HBA) utilities to change the node identity.

Figure 3 shows an example of how to implement a spoofing attack in either a Fibre Channel or iSCSI SAN. In this example, the node WWN of the FC HBA is modified to match another HBA node WWN that is authenticated to log in to a storage port. Similarly, in iSCSI SANs, the IQN value is easily changed through native HBA utilities. Fortunately, this type of attack can be mitigated with the implementation of appropriate security mechanisms and configuration changes. (Refer to "Secure SAN architecture, components, and mechanisms" on page 24.)

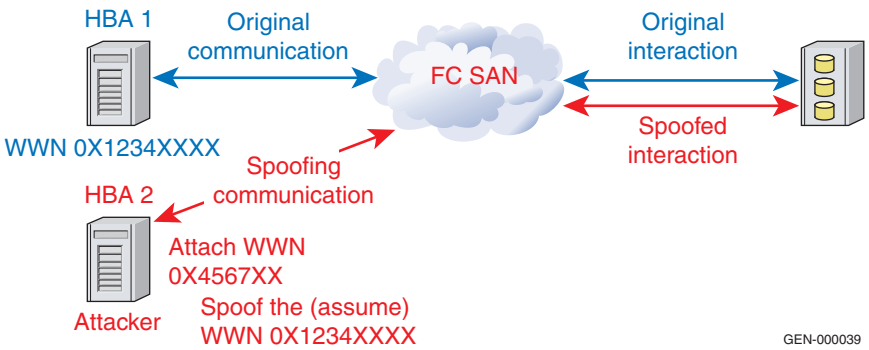


Figure 3 Spoofing in a SAN

Denial-of-service (Disruption)

A denial-of-service (DoS) attack is a deliberate act to prevent an authorized user from accessing data. Limitations of FC and iSCSI SAN protocols can be exploited in order to bring down the network. For instance, the network interface can be flooded with undesired traffic or conflicts can be created that cannot be resolved by the SAN, thereby preventing access to data.

Figure 4 shows an example of DoS in a SAN. In this example an attacker can, after snooping the transaction between the two nodes, issue a port discovery (PDISC) to change the exchange service information to a node. As a result, the service between the two nodes is disrupted.

Other DoS attacks can also precede data theft or destruction. For example, in an iSCSI SAN two iSCSI node names, one authorized and one spoofed, can try to gain access to the same target LUN. Some storage targets will drop the authorized node and grant access to the spoofed node thinking that this is a failover attempt. Other implementations do not allow two node names to connect to the same LUN at the same time, resulting in DoS. Without proper precautions, an unsuspecting administrator might elect to troubleshoot by rebooting the authorized node, thereby giving sole read and write permissions to the spoofed node for the duration of the reinitialization.

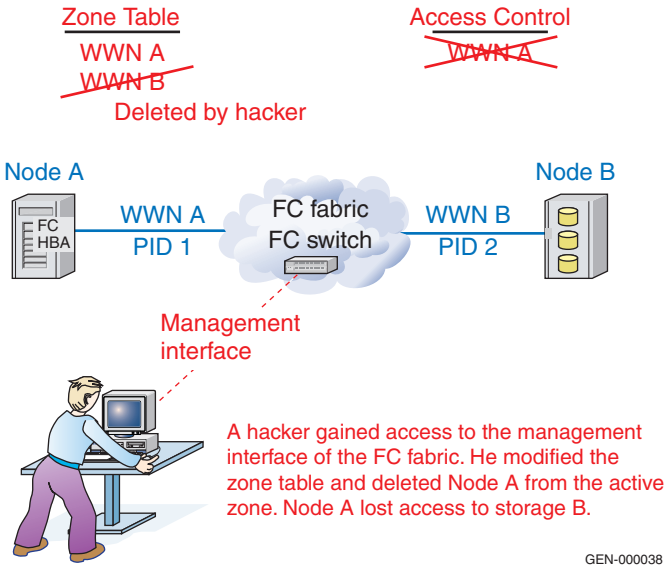


Figure 4 **Denial-of-service attack**

Secure SAN architecture, components, and mechanisms

Securing viable SANs with multiple servers, storage targets, distance extension components, and administrators is complex. Designing security into SANs requires cross-functional investigation, quantification of risks, and breach of security contingency planning before the design can be implemented and tested to satisfy business and regulatory requirements.

There is no single comprehensive security solution. Many argue that this is a result of security features not being accounted for in the early development of industry standards. Initial iSCSI SAN and FC SAN protocols have inherent weaknesses with authentication and authorization. SAN vendors have implemented partial solutions to address security issues; however, many of these solutions are not compatible with emerging standards or interoperable with other existing vendor security offerings.

Standards committees, such as the Fibre Channel Security Protocol (FC-SP) and the IETF IP Security Working Group (IPSEC), are making progress in strengthening the security aspects of these protocols. The FC-SP standard describes the protocols used to implement security in Fibre Channel fabrics. This standard includes the definition of protocols to authenticate Fibre Channel entities, protocols to set up session keys, protocols to negotiate the parameters required to ensure frame-by-frame integrity and confidentiality, and protocols to establish and distribute policies across a Fibre Channel fabric. IPSEC standards are similar, as evidenced in the standardized security elements found in IPv6.

The evolution of SANs includes both the Fibre Channel and IP storage transport protocols. Security vulnerabilities for FC and IP SANs are similar; however, their solution mechanisms and effectiveness may differ. Table 1 on page 26 presents some of the mechanisms available to mitigate security risks.

This section discusses:

- "SAN architectures" on page 25
- "Security components" on page 26
- "Security mechanisms" on page 26

SAN architectures

Secure SAN architectures usually require that multiple security domains, or zones, be implemented and that these security zones be formally documented and controlled to meet regulatory auditing requirements. Security zones can exist between servers and switches, between switches, between SAN management systems and switches, and between administrators and access control management systems.

Figure 5 depicts such security zones. The boxes indicate some of the relevant security components and mechanisms that can be deployed to control security in each of these security zones.

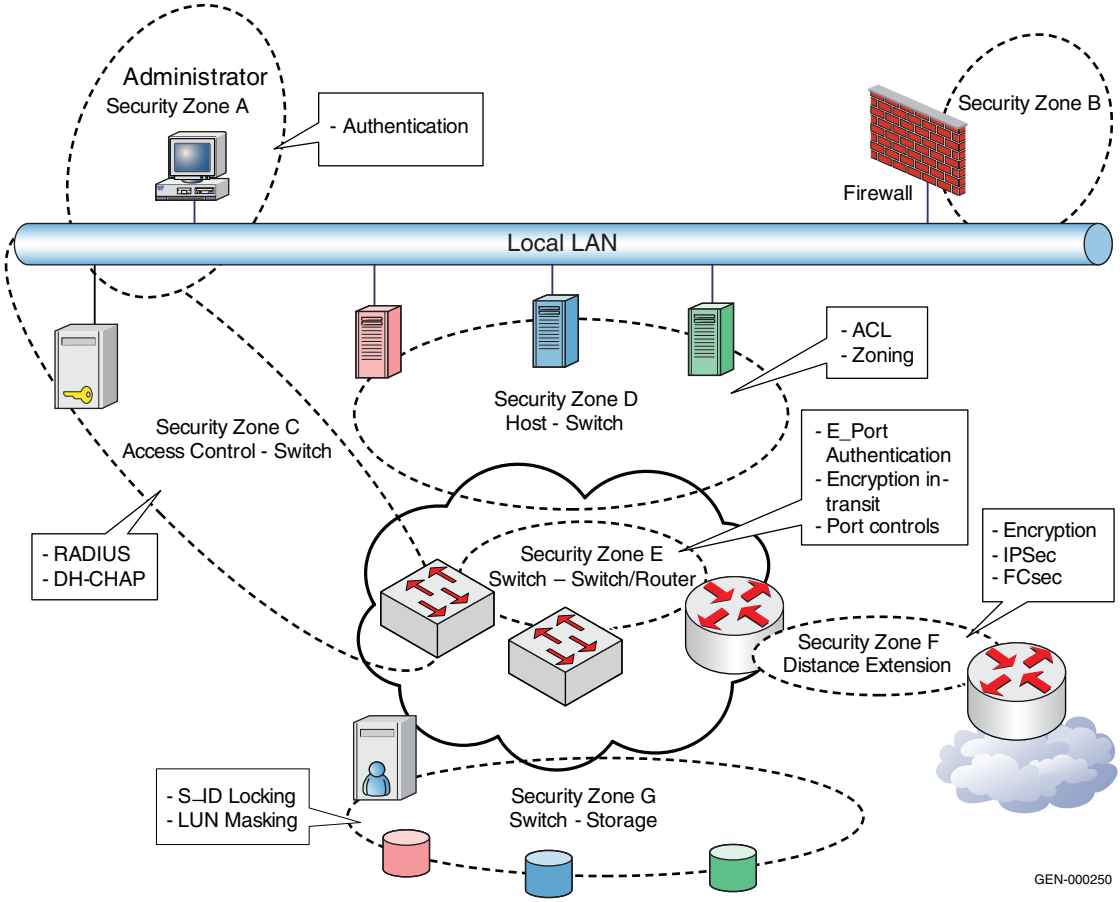


Figure 5 **Security zones**

Security components

Authentication, authorization, auditing (AAA), data integrity, availability, and encryption are frequently referred to as *components* of a secure implementation. Within SANs, these fundamental components of security are implemented through various *mechanisms* (described next) to provide secure data access, secure management, and data integrity.

Security mechanisms

Available mechanisms that promote a secure SAN include:

- Access control
- Zoning
- LUN masking
- Port Binding
- Management keys
- Protocols
- Encryption
- Physical access control mechanisms

These mechanisms can vary by topology, vendor, and business needs. Table 1 lists some of these component mechanisms. Specific design considerations and vendor-specific implementations for some of these mechanisms are presented in "Building secure SANs" on page 52.

Table 1 Secure SANs component mechanisms (page 1 of 2)

Security category	FC SAN mechanisms	IP SAN mechanisms
Availability	BB_Credit	QoS
Authentication	SLAP DH-CHAP FCAP FCPAP IKEv2-AUTH CT-Authentication (FC-GS-4)	CHAP KBR RADIUS TACACS+ Kerberos SRP

Table 1 **Secure SANs component mechanisms (page 2 of 2)**

Security category	FC SAN mechanisms	IP SAN mechanisms
Authorization	Hard port-based zoning Soft port-based zoning LUN Masking Port controls Port Binding	iSCN LUN Masking VLAN Tagging Port controls
Auditing	ACL SSH SSL	ACL SSH SSL
Encryption	FC-SP (ESP) In-transit Algorithms At-rest Algorithms	IPSec In-transit Algorithms At-rest Algorithms
Integrity	MD5 hash SHA-1 hash	IPSec (AH) MD5 hash SHA-1 hash

The following sections provide further information and suggestions to secure data access, SAN management, and data integrity:

- "Securing data access" on page 27
- "Securing SAN management" on page 30
- "Securing data confidentiality and integrity" on page 33

Securing data access

Securing access to data within a SAN includes authentication and authorization components to control access to data as well as access to SAN management functions. Mechanisms can be deployed in zones that exist at the ends of the data path or in zones that provide the interconnection in the network (for example, FC fabric switches, TCP/IP network switches, routers). Access control lists, zoning, LUN masking, binding, and port controls represent such mechanisms. Each of these mechanisms are further discussed in this section.

Access control

Implementing access control includes securing access to the management console, maintaining access control lists, and securing access to ports within a SAN. Access control can be complex and may fail to meet auditing requirements in the absence of documented

policies. Moreover, without strong authentication and encryption in the data path, access control is not as secure as one would expect.

Secure access to a management console should include restricting the IP address for the management application. Switch management should be done only from secure networks. Remote access should be provided through a secure tunnel network, such as a Virtual Private Network (VPN). Secure access can be improved by implementing two-factor authentication that uses an additional security component, such as a smart card, so that the access is only given when a user id is authenticated against a password and a key. Other user login measures should include policies for password generation/renewal, key pass phrase requirements, and hash time limitations.

Access control lists (ACL) and policies provide authorization for access to SAN resources. This form of authentication is known as Role-Based Access Control (RBAC). Different access levels provide an additional level of accessible features and functions. Moreover, RBACs may be used to separate data access from data management. User login permissions for different access levels, such as equipment manager, security officer, recovery officer, network admin, storage admin, and so on, should be set.

Physical access to network ports should also be secured. Key card access to restricted areas, as well as port controls that are available from Fibre Channel and IP switch vendors, should be utilized.

Zoning

Fibre Channel switch zoning is analogous to IP-based switch VLANs. Nodes are segmented into zones and given access to other nodes within the same zone. Zone members have any-to-any connectivity within the zone, whereas non-zone members have no connectivity. Zoning is a technology that uses hardware or software to create segments in a single SAN fabric.

Vendors have also developed technologies like VSAN, LSAN, and VLAN tagging that use hardware and software to create separate SAN fabrics, preventing even zoning to be performed across fabrics unless routed. The purpose is to allow the grouping of a set of nodes. As a result, access is limited to a certain set of nodes (either initiators or targets) whereby both intentional and unintentional access to data is restricted. Other advantages of zoning include lowering the effects of State Change Notifications (SCN) and broadcasts, as well as controlling traffic of particular nodes for performance.

Zoning can be *soft* or *hard*. *Soft zoning* is based on the nodes World Wide Name (nWWN) regardless of the physical port on the switch to which it is connected. Soft zoning does not perform any filtering of frames among zones; it merely restricts routing information from being passed to unauthorized zone members. One significant advantage of soft zoning is its flexibility and ease of management. For example, with soft zoning if a node must be physically moved and plugged in to a new port, its zone membership stays intact. A significant security disadvantage is that a malicious node could spoof any given nWWN on the fabric and access data located on a LUN to which it normally would not have access.

Hard zoning, using physical switch ports for zone membership, is a more secure zoning method. Not only is routing information not passed to unauthorized zone members, but frames are filtered to ensure only authorized zone members can communicate. WWN spoofing and route-based attacks would be defended. Hard zoning security benefits come at the cost of SAN management.

LUN masking

Storage device logical unit numbers (LUNs) connected to a SAN are made visible to host devices. LUNs can be presented to one or more hosts. The ability to allow a given host to see a LUN is referred to as *LUN masking*. LUN masking is a common method of controlling host to LUN access. LUN masking can be implemented at the host node, within the switch, or at a storage node.

EMC recommends implementing LUN masking at the storage node in combination with hardware-enforced WWPN zoning. This affords the authorized storage administrator control while protecting the LUNs from spoofing attempts.

Initiator access to storage LUNs can be controlled by such LUN masking tools as EMC® Symmetrix® Device Masking and CLARiiON® Access Logix™. Symmetrix systems implement a Volume Configuration Management Database (VCMDB) so that a HBA can only access a particular set of LUNs. CLARiiON implements a storage group for the same purpose. In both implementations, hard zoning should be used to circumvent malicious or accidental changes to the LUN masking table.

Fabric Binding

Fabric Binding mechanisms allow only authorized switches to join or disrupt the current fabric. ISLs are only enabled between specified switches in the fabric. Membership data is used to ensure that the list

of authorized switches is identical in all switches in the fabric. Fabric Binding helps to ensure that unauthorized switches are segmented from the fabric and that only an authorized switch is merging into the expected fabric.

Port Binding/S_ID lock down

Port Binding is a SAN security mechanism that uses switch software or hardware to associate between a physical port ID and host WWN. This association will mitigate snooping attempts. A malicious node on a bound port attempting to spoof another node's WWN would not be able to connect to another node since the spoofed WWN is not associated with the port ID. When using Port Binding, all nodes need to be bound to their specific port since nodes that are not bound can still spoof.

A similar security mechanism to mitigate spoofing in shared storage port configurations is to use a Source ID (S_ID) Lock Down feature, like the one available on Symmetrix systems. By mapping the S_ID with the WWN of a HBA into the VCMDB, only a HBA physically connected to a specific switch port is allowed to log in to the storage port. Once a S_ID is locked down, a spoofed HBA WWN cannot log in. Further, if a spoofed WWN is already logged in, that storage user loses all access from that HBA.

Port controls

Port security controls complement Fabric Binding by helping to prevent unauthorized access to a switch. Port controls, such as port locking and port-type locking, can help to protect the overall SAN infrastructure. Port locking persistently prohibits an unused port from joining a fabric. Port-type locking forces a G_Port to act as only an F_Port, N_Port, or E_Port. Implementing such controls limits the expansion of the fabric and lowers the risk of a physical security attack.

Securing SAN management

SAN management consoles are primary targets for attackers. Risks include usage of clear text management protocols, weak username and passwords, un-segmented communication networks, and shared accounts. Administrators should deploy strong authentication and authorization mechanisms to secure SAN management. Implementation decisions are necessary to secure SAN management functions while balancing business needs for accessibility and performance.

Authentication is also not limited to storage administration. It is also important when a switch connects to another switch through ISL. During the ISL process, crucial information regarding the fabric is exchanged (like zoning information) and can be compromised if a rogue switch manages to join the fabric. This can lead to either corruption of the fabric or leakage of information.

Without authentication there is implicit trust that every switch connecting is trusted. While this facilitates ease of administration, it also introduces a serious flaw in the design. This section discusses switch authentication and management authorization.

Switch authentication

The main authentication protocols proposed are:

- DH-CHAP (Diffie-Hellman Challenge Handshake Authentication Protocol)
- FCAP (Fibre Channel Authentication Protocol)
- FCPAP (Fibre Channel Password Authentication Protocol)

The Fibre Channel Security Protocol (FC-SP) specification provides for host-to-switch and switch-to-switch authentication. The FC-SP specification defines authentication protocols as being secret-based, certificate-based, or password-based.

Under secret-based protocol, DH-CHAP allows for authentication between Fibre Channel switches, but it can also allow for client nodes and switches or storage controller nodes and switches exchanges. It is essentially a CHAP protocol that is augmented with an optional DH algorithm for shared key exchange.

The authentication involves two entities:

- Authentication initiator
- Authentication responder

For a DH-CHAP initiator to be authenticated, it must provide a unique name that is tied to a secret that is either known or obtained from a centralized RADIUS or TACACS server. To enable secured processing after authentication, the optional DH algorithm can be used as the means to generate the session key linked to the Security Association Management Protocol.

Remote Authentication Dial In User Service (RADIUS) is a security protocol in network environments and is often embedded in switch and router components. User profiles are maintained in a database that remote components can share to authenticate and authorize

access to the network. Provisions are typically made for local or centralized authentication through RADIUS or TACACS servers.

Digital certificate-based implementations allow responders or initiators to trust a certificate-based infrastructure whereby the components are certified by a trusted Certificate Authority. The certificates generated by the CA implies trust that each entity with a public-private key pair can be used to mutually authenticate with other certified entities through the FCAP protocol. Certificate Authorities need not be online, although there might be a requirement for a directory service like LDAP to be online for the public key infrastructure to work. FCAP is based on the PKI infrastructure with the added benefit of certificate validation. Upon successful authentication, the shared session key can be calculated from the exchanged random nounce (number used once), DH group parameter, and hash value.

Password-based authentication protocol establishes a security relationship by having zero-knowledge password proof of the password-based credential material of other entities. This means that the protocol is resilient to password guessing. Entities may use this credential material to mutually authenticate with other entities using the FCPAP protocol. It is based upon the Secure Remote Password Protocol (SRP). The protocol authenticates using a random salt, a verifier that is computed using the salt and password together with a hash function.

Under the FC-GS-4 specifications, an authentication protocol for communication, the Common Transport Authentication (CT), exists for performing in-band management purposes,.

Similarly, iSCSI SANs' authentication mechanisms include MD5 CHAP, SRP, and iSCSI Kerberos. In addition to these authentication mechanisms, management communication channels should utilize encrypted protocols such as SSH and SSL.

FC and IP switch vendors provide these various security mechanisms to authenticate switch management and merge activity. Authentication in a secure fabric verifies the identity of a new switch before the switch is allowed to join the fabric and also ensures the new switch is joining the expected fabric. Properly implemented, these mechanisms only allow switches that are authenticated and authorized to join a fabric. Since many of these mechanisms have vendor-unique features, secure switch interoperability needs to be considered by the administrator in mixed-vendor environments.

Management authorization

Security management addresses authentication and authorization, as well as the logging of operations for auditing. Authentication and authorization of management access to the network needs to be implemented and enforced with policies and rules. Two-factor authentication for management access is a best practice, as is segmenting the management node from internal communications networks. Separation of management duties should be implemented by establishing ACLs and Role Based Access Control (RBAC), whereby levels of access and functions that a user can perform are controlled and monitored.

Note: Every authenticated user should *not* be granted authorization for all SAN management functions.

Securing data confidentiality and integrity

Data confidentiality includes the encryption of data to prevent reading by others, while data integrity verifies that data sent has not been altered. Securing data confidentiality and integrity can be done with physical security and with encryption, each discussed in more detail in this section.

Physical security

Controlling physical access to SAN equipment is a basic security feature. Mechanisms to implement physical security consist of access cards, locks, gates, in-circuit video surveillance, guards, and armored mobile transport. Policies to physically secure multiple data sites and the network between distributed sites are difficult to implement and costly. The industry trend is leaning towards building security into the SAN to avoid sole reliance on perimeter security deficiencies and its associated costs.

Encryption in the SAN

Zoning, ACLs, and authentication security tools address storage access, but confidentiality of data is not ensured. *Encryption* and *decryption* provide a level of security beyond the access-control or perimeter-guards. These mechanisms create another layer of security to better prevent an unauthorized entitle from reading the intended data. Data can be encrypted "in-transit" or when "at rest." The use of encryption is dependent on how and where it is deployed.

Encryption basics

Encryption refers to a method that transforms data in plain-text from a legible form into a non-legible form, otherwise known as *cipher-text*, as shown in Figure 6. This is accomplished through the use of cryptographic algorithms. The reverse process is called *decryption*.

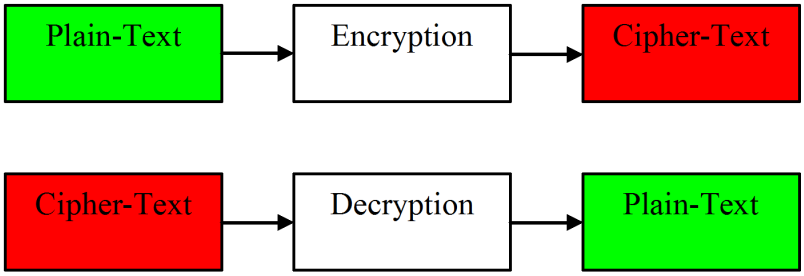


Figure 6 **Encryption process**

Currently, the cryptographic algorithms used are often publicly known, so in order to provide data confidentiality some *other* variable used in the algorithm should be kept secret. This secret variable is known as the *encryption key*.

These keys contain random bits. How randomly mixed these bits are depends on how large the available keyspace in the algorithm is to accommodate it. The algorithm contains the keyspace that specifies the size of the key that it can handle. A good, strong encryption will consider the algorithm, secrecy of the key, length of the key, initialization vectors, and how they all work together within the cryptosystem.

Symmetric key encryption algorithms, such as Blowfish, Advanced Encryption Standard (AES), and Data Encryption Standard (DES), work with a single secret key shared between sender and receiver for both encryption and decryption, as shown in Figure 7.

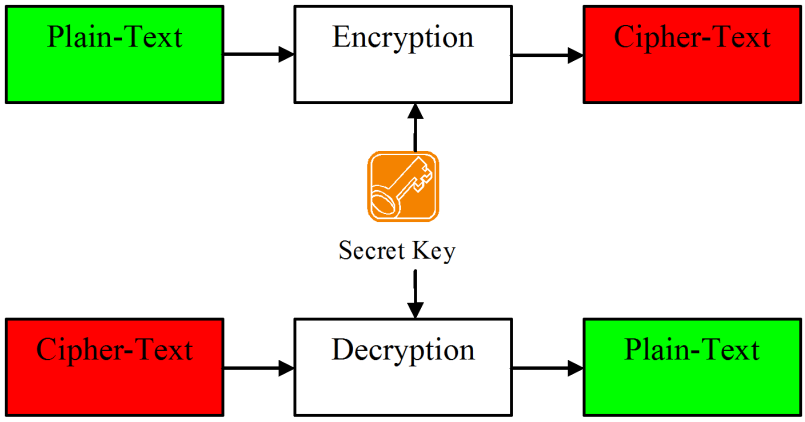


Figure 7 **Symmetric encryption example**

AES based on FIPS 197 has several modes of operation of which five are approved by NIST:

- ECB — Electronic CodeBook mode
- CBC — Cipher Block Chaining mode
- CFB — Cipher FeedBack mode
- OFB — Output FeedBack mode
- CTR — Counter mode

These modes of operation essentially spell out how the AES algorithm performs the number of required rounds for the encryption and decryption operation. While these modes of operations are suitable for most security implementations, they are less suitable for storage devices, which need to take into consideration the dynamics of a disk or tape's use of random or sequential data reads and writes in a sector.

The IEEE P1619 working committee proposes another mode of operation in its standard architectures for encrypted shared storage media suitable for disk:

XTS - Tweaked CodeBook mode with CipherText Stealing

The committee formed IEEE P1619.1, which proposes yet another mode of operation in its standard for authenticated encryption with length expansion for shared storage media suitable for tape backups:

GCM - Galois/Counter Mode

In *asymmetric* encryption schemes, such as RSA algorithm, the scheme creates a "key pair" for the user: a public key and a private key, as shown in Figure 8. The public key can be safely published online for senders to use as the encryption key to encrypt data meant for the owner of the public key. Once encrypted, the cipher-text cannot be decrypted except by the entity which has the private key of that key pair. This algorithm is based on the two keys working in conjunction with each other.

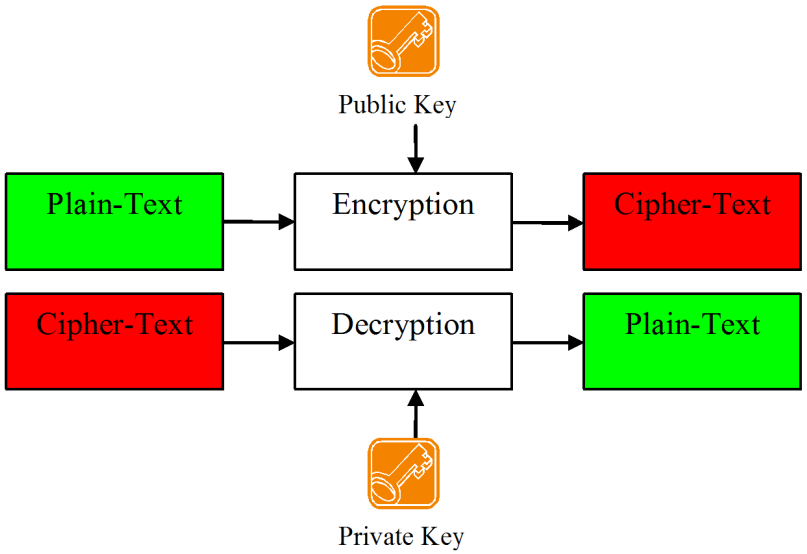


Figure 8 **Asymmetric encryption example**

Asymmetric encryption is considered one step more secure than symmetric encryption because the decryption key can be kept private. The caveat of asymmetric encryption is that the operation is computationally more intensive as compared to symmetric encryption.

Secure key exchange

It is quite common to secure communications by making use of the security of asymmetric ciphers together with the speed of symmetric ciphers.

Example 1

Assume two entities, Alice and Bob, wish to communicate, but using public key encryption to secure the current communication session will incur too high of an overhead. There is the issue of key reuse. Keys can be thought of as expendable items with a limited lifetime dependant on its usage. Such frequent use will greatly "wear" down a key and a new public–private key pair will need to be reissued. This brings about another dimension of key management issues that would be best to avoid.

A private key encryption is more suited for providing session confidentiality. Keys can be created and destroyed, or archived, after each session, thereby maintaining a more secure communications channel. However, the dilemma still remains to distribute the session key securely, efficiently, and effectively.

One alternative is using an *out-of-band* method, which communicates the secret through a channel other than the channel that the secret is meant to secure. However, how secure using out-of-band channel really is depends upon many factors.

A possible way to solve this dilemma is to make use of *public key cryptography* to secure the session key and distribute it across the network to the peers. In the following scenario, we will assume that Alice and Bob have already established their own public–private key pair.

1. Alice and Bob wish to communicate privately.
2. Alice generates a secret key for use in the session communication.
3. Alice then uses Bob's public key to encrypt the secret key.
4. Alice sends this encrypted secret key across the network.
5. Bob retrieves the encrypted secret key.
6. Bob uses his private key to decrypt the encrypted secret key.
7. Bob now has the secret key he will use to communicate with Alice, and vice versa, for the session.

Example 2 The use of public key cryptography can be further expanded to ensure integrity and non-repudiation through the use of Certificate Authority signing of public keys with the use directory services for publishing the public keys. This is essentially the basis for Public Key Infrastructure (PKI), which needs these other components within the infrastructure to properly implement.

1. Alice and Bob wishes to communicate privately and trusts that the other party is who they say they are.
2. Alice generates a secret key for use in the session communication.
3. Alice creates a hash of the secret key and encrypts it by using her private key to ensure the integrity of the secret key.
4. Alice then retrieves Bob's public key from the public directory service.
5. Alice is assured of Bob's identity by trusting the Certificate Authority's signature on his public key. (It is assumed Alice has explicit trust of the CA.)
6. Alice then uses Bob's public key to encrypt the secret key together with the encrypted hash value.
7. Bob retrieves the encrypted payload.
8. Bob uses his private key to decrypt the encrypted payload.
9. Bob will then obtain the secret key and the associated encrypted hash.
10. Bob then retrieves Alice's public key from the public directory service.
11. Bob is assured of Alice's identity by trusting the Certificate Authority's signature on her public key. (It is assumed Bob has explicit trust of the CA.)
12. Bob uses Alice's public key to decrypt the encrypted hash value.
13. Bob will verify the secret key by calculating its hash value against the decrypted hash value.
14. Only on verification of the hash does Bob have confidence that the secret key originated from Alice and that it was not modified on transit.
15. Bob now has the secret key that he will use to communicate with Alice, and vice versa, for the session.

Other methods of key exchange

Other methods of key exchange are through the use zero of knowledge cryptographic protocols that employ Diffie-Hellman (DH) or Secure Remote Password (SRP). These protocols rely on the complexity of the discrete logarithm problem making it safe to use as is it very difficult to break within the short span of time within a session.

Note: Encrypted data does not protect against a malicious user's ability to destroy encrypted data or against denial of service attacks. Backup policies and additional security mechanisms need to be implemented even when data is encrypted.

Key encryption management concerns

A major consideration when implementing encryption is the management of the keys used for encryption and decryption. Lost keys, restores, performance, and encryption component failovers cannot be ignored to meet availability needs for encrypted data. Key management systems need to be established to ensure that data can be recovered when encryption is used improperly and to mitigate the effect of lost or stolen keys.

SAN encryption types

In network storage security, we are concerned about data residing in the target devices and the path of travel from target to initiator. Data can be encrypted "in-transit" or when "at rest." The use of encryption is dependent on how and where it is deployed.

Encryption of *data-in-transit* incorporates components that provide a secure communication tunnel in the middle of the session between the initiator-target or in the ISL between two switches. The iSCSI implementation in Microsoft Windows allows the host to embed IPsec encryption with the iSCSI initiator software. In FC SANs, FC-SP has provision for encryption of data-in-transit although this is not as mature as IPsec. Cases where data-in-transit might be used include:

- Protection from network sniffers
- Assurance of business continuity remote replication solutions

If data security is a concern behind the "perimeter," then one solution is to encrypt the data on the storage media. This is known as encryption of *data-at-rest*. Encryption of data-at-rest allows data to be kept confidential at its final destination and remain in its encrypted form.

Cases where data-at-rest are required through regulation or company confidentiality policies include:

- Backup to tape
- Removal of disk for repair
- Protection of data between and in disaster recovery sites
- Data extracts sent to service providers and partners

Other cases where the intention is to prevent breach of unauthorized access include:

- Shared/consolidated storage used by numerous groups
- Protecting data from insider theft (such as employees, administrators, contractors, janitors, etc.)
- Protection of application/executables from alteration

Implementation of SAN encryption

Encryption can be applied to different areas of the SAN depending upon the needs of the organization. The different areas by which encryption can be implemented are categorized and discussed in the following sections and illustrated in Figure 9 on page 41:

- "Application level" on page 41
- "Host level" on page 44
- "Network level" on page 46
- "Device level" on page 48

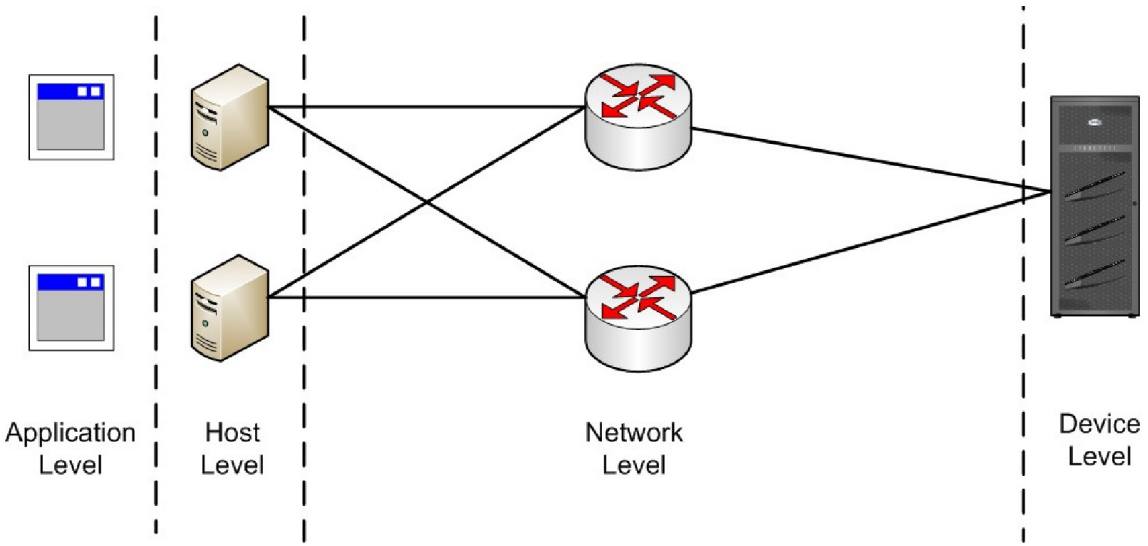


Figure 9 **Encryption levels**

Application level

Perhaps the greatest control over information can be exercised from where it originates, the application. The application has the best opportunity to classify the information and manage who can access it, during what times it can be accessed, and for what purpose.

If the administrator has concerns or perceives risks over the information at all levels in the infrastructure, it makes sense to begin with applying security at the application level and then work down through the other levels. Adding encryption at the application level allows for granular, specific information to be secured as it leaves the application. For example, a database could encrypt specific rows/columns of sensitive information (for example, Social Security numbers or credit card numbers) while leaving less sensitive information unencrypted. Attempts by others to breach security would yield only useless information.

Encryption at the application level provides security from access at both the operating system level as well as from other applications on the server. The application still needs to provide user authentication and authorization to guarantee that only those authorized can access the application and the data; otherwise, application-based encryption provides no additional security benefit.

Challenges

The following are some drawbacks to encrypting at the application level:

- Encryption is done on a per-application basis. If there are multiple applications needing encryption, each would have to handle the task separately, creating additional management complexities to ensure that all confidential data is protected.
- Application-level encryption solutions are typically software-based. Encryption is a CPU-intensive process and will compete with normal operating resources on the server. In addition, the encryption keys will be stored in dynamic, non-volatile memory on the server. If a hacker were to break into the server and find the keys, the information can be decrypted. Externalizing the encryption engine or key manager may address these issues, but at the expense of additional costs. An external key manager also enables clustered applications to share key information across nodes and geography (provided that each node can supply a secure channel from the server to the key manager). If FIPS 140-2 compliance is a requirement for the encryption solution, an external appliance is typically used.
- Application-based encryption presents challenges in the area of re-keying. Any effort to re-key the data (to protect the integrity of the keys) has to be done by the application. The application will need to read and decrypt the data using the old key and re-encrypt and rewrite to disk using the new key. The application will also have to manage old and new key operations until all the previously encrypted information is re-encrypted with the new key. This will most likely be done while the application is handling normal transactions, presenting resource contention issues.
- The introduction of business intelligence solutions in the enterprise provides yet another challenge. Encryption at the application level exposes only encrypted information to other applications (including backup) and devices in the stack. Any attempt to perform analysis on the data is useless, as patterns and associations are lost through the randomization process of encryption. To accomplish any analysis, the business intelligence applications will need to be associated with, and linked to, the application performing encryption to allow for a decryption of data at a level outside of the application. This introduces a possible security risk.

- Application-based encryption must also account for variable record lengths. Encryption schemes must pad data up to its block size to generate valid signatures. Depending on the implementation, this may require some changes to application source code.
- Application-based encryption does not take into account the impact on replicated data. Any locally-replicated information at the storage layer (a clone), does not have visibility into the application and the keys; nor does the application have visibility into the replication process. Key management becomes more complex. In addition, compression in the WAN is impossible for remote replication of the encrypted information, causing WAN capacity issues.
- End-user activities, after data is converted to clear text, are potentially the highest security risk for organizations.

EMC solutions

EMC helps address information protection at the application level, both in providing application development support with the RSA® BSAFE® tools and the RSA Key Manager and by delivering application encryption with the following solutions:

- **Backup**

  EMC NetWorker®, Retrospect®, and Avamar® feature native encryption.

- **Archiving**

  EMC EmailXtender® encrypts all user messages in local cache.

- **Unstructured content**

  EMC Documentum® Trusted Content Services provides file store encryption to secure content in repositories, and Documentum Information Rights Management encrypts documents to control viewing, printing, copying, and other activities once documents have left the repository.

  RSA File Security Manager encrypts laptop, desktop, and server files.

- **Database**

  RSA Database Security Manager encrypts database objects within IBM, Oracle, Microsoft, Sybase, and Teradata environments.

Host level

Encrypting at the host level provides very similar benefits and trade-offs to application-based encryption. At the host level, there are still opportunities to classify the data, but on a less granular basis. Encryption can be performed at the file level for all applications running on the host. However, there are options for a host-based adapter or software to provide encryption of any data leaving the host as files, blocks, or objects.

One example for host-based encryption operating at the logical unit level (blocks) is EMC PowerPath®. As with application-level implementations, the operating system must still provide user authentication and authorization to prevent against host-level attacks. If these strong access controls are absent, host-level encryption will provide no additional security benefit (aside from protection against loss or theft of media).

As encryption is performed at the host level, the data can be of variable record length. Similar to the application-based approach, the encryption solution can add information to the encryption payload to allow for a digital signature or cryptographic authentication. This would prevent a "man-in-the-middle" from substituting bad packets for the good encrypted packets from the host. PowerPath performs block-based encryption at the Logical Unit level with no added data to the payload.

If implemented correctly and integrated with the encryption solution, host-level encryption can provide some process authorization granularity, managing which users should be allowed to view plaintext data. At the host level, encryption can be done in software, using CPU resources to perform the actual encryption and either storing the keys in memory or offloading the keys to specialized hardware.

- For an accelerator card approach, encryption is done as a look-aside operation independent of the transport. This provides flexibility for host connectivity, but increases the memory and I/O bus load in the system.
- Offloading the keys involves the use of an HBA or an accelerator card resident in the host to perform the actual encryption of the data. In the case of the HBA, the encryption can be performed in-band and is dedicated to the particular transport connection from the host; that is, Fibre Channel.

In either case, the host software would control the connection to the key manager and management of the keys.

There may be a need in the enterprise for the host-based encryption solution to support multiple operating systems, allowing for interoperability across systems or consistency in the management domain, something to consider when evaluating solutions. In addition, when encryption is implemented at the host level there is the flexibility of being storage- and array-independent, allowing for support of legacy storage with no new hardware needed.

Challenges

The following are some drawbacks to encrypting at the host level:

- Host-based encryption does present a challenge when coupled with storage-based functionality; that is, replication. If replication is employed underneath the host encryption level, the host implementation must have the ability to track replicas and associate encryption keys, eliminating the need for users to manually manage the replication and encryption technology. As host encryption supplies encrypted data to the array, remote replication would transmit encrypted, uncompressible data. This would severely impact WAN performance.

  However, PowerPath, a multi-pathing I/O device level application, provides mechanisms to deal with both the cross operating system interoperability and replication issues. As PowerPath provides consistent functionality through releases on Solaris, Windows, Linux, AIX, and HP-UX, there is a single implementation of encryption and key management, independent of the operating environment.

  PowerPath has developed a unique approach to managing replicas with encryption, allowing for coordinated key management between source and replicated volumes independent of user intervention.

- As with application-based encryption, business intelligence solutions in the enterprise pose additional complexities. Encryption at the host level will expose only encrypted information to other hosts and devices in the stack, introducing the same challenges with analysis as those described in "Challenges" on page 42.

EMC solutions

EMC offers solutions to address information protection at the host level, both in providing application development support with the RSA BSAFE tools and the RSA Key Manager as well as delivering host encryption with:

- **Backup**

  NetWorker, Retrospect, and Avamar

- **Unstructured content**

  RSA File Security Manager

- **Host-based**

  PowerPath features block-based encryption on Logical Units

Network level

If the threat in the enterprise is not at the server, operating system, or application level, but rather at the network or storage level, then a network-based appliance approach for encryption may work best.

This approach is operating system-independent and can be applied to file, block, tape, Fibre Channel, iSCSI, or NAS data. Encryption and key management are handled entirely in the hardware and runs at the wire speed of the connection. The appliance presents an "unencrypted side" and "encrypted side" to the network. Encryption can be designated on a per block, file, or tape basis and the keys are maintained for the life of the data. Appliances available today are typically FIPS 140-2 level 3 validated.

There are two implementations for a network-based appliance design:

- Store-and-forward
- Transparent

The *store-and-forward* design appears as storage to the server and as a server to the storage, and supports iSCSI, Fibre Channel, SAN, NAS, and tape. An I/O operation comes to the appliance and is terminated. The data is encrypted and then forwarded to the destination storage device. This approach adds latency so ideally some form of "cut-through" needs to be offered to minimize the impact of the device for non-encrypted traffic. In addition, to appear as both server and storage, the store-and-forward appliance either needs to spoof the identities of the attached devices or rely on robust security practices to counteract the attempts to circumvent the appliance.

While there may be a latency penalty for encrypting data through the appliance, the store-and-forward-based design has the benefit of allowing the attached-storage devices to be re-keyed in the background. This is performed with no disruption to host operations as all I/O operations to the storage are handled independently of the host. There may still be some performance impact to the re-keying process, depending on the I/O load on the encryptor.

The *transparent* approach provides a flow-through model for the data being encrypted, supporting Fibre Channel SAN and tape. The appliance inspects SCSI headers as the data flows through the appliance and encrypts *only* the data payloads that match preset source/destination criteria in the appliance configuration. The latency associated with this approach is minimal. The transparent design does, however, have a drawback when the encrypted data needs to be re-keyed. Unlike the store-and-forward design, the device is essentially transparent in the data flow, requiring the host to perform the reads and writes required in re-keying the encrypted data. This process can be done by a separate host agent and can be performed while normal operations are in process.

Challenges

The following are some drawbacks to encrypting at the network level:

- For block-based implementations, the size of the encrypted data cannot increase. This means no additional information can be added to the encrypted payload (for example, a digital signature). This is not true for file or tape-based encryption where the record information may be variable. As previously mentioned, the IEEE is working to provide standards for encrypting block data-at-rest, in IEEE P1619.
- There may be a need in the enterprise for the encryption to support multiple operating systems, allowing for interoperability across systems or consistency in the management domain. When encryption is implemented at the network level, there is the flexibility of being storage- and array-independent, allowing for support of legacy storage, but at the cost of adding new hardware. Hardware in this case is added in increments of ports, typically two at a time, adding to the power, package, and cooling issues currently facing enterprises today. In addition, adding appliances in these increments can add complications in managing additional devices in the enterprise.

- Network-level encryption does present a challenge when coupled with storage-based functionality such as replication. If replication is employed underneath encryption at the network level, the implementation must have the ability to track replicas and associate encryption keys, eliminating the need for users to manually manage the replication and encryption technology. As network-level encryption supplies encrypted data to the array, remote replication would transmit encrypted, uncompressible data. This severely impacts WAN performance.
- Network-level encryption does not take into account the impact on replicated data. Any locally replicated information at the storage layer (a clone), does not have visibility into the network device management and the keys nor does the network device have visibility into the replication process. Key management can become more complex and more manual. In addition, compression in the WAN is impossible for remote replication of the encrypted information, causing WAN capacity issues.
- There are implementations moving to use data integrity features as part of the protocols. Encryption in the network level would encrypt both the data and the data integrity, resulting in mismatches at this level of checking performed at the arrays.

EMC solution

EMC offers the following solution to address information protection in the network, through partnership with Cisco:

Cisco Storage Media Encryption (SME) or Connectrix® MDS provides encryption of data-at-rest as a service with its switches with proper key management facilities.

Device level

Encryption at the device level, (array, disk, or tape) is a sufficient method of protecting sensitive data residing on storage media, which is a primary security risk many organizations are seeking to address. Encryption at the device level can be at the following sublevels, each discussed briefly in this section:

- "Array-level encryption" on page 49
- "Tape-level encryption" on page 51

All data written to the device is encrypted and stored as such and then decrypted when read from the device. Encryption at this level is application- and host-independent, and can also be transport

independent. When addressing media theft, the granularity for encryption, and keys, can be at the disk or tape level.

Array-level encryption

There are a number of design points for encryption in the array, that is, at the **disk drive** or **controller** level, each discussed briefly, next. Design considerations for encryption include the interfaces to the array, software support, performance, FIPS validation, key management, and encryption object granularity, to name a few. The intent is to have the encryption implementation transparent to the hosts attached, while protecting the removable media. The connected hosts may not be knowledgeable of the encryption implementation but may be with respect to management and performance. All aspects of the design must be considered.

- Disk drive encryption

  One possible approach is to implement the encryption in the **disk drive**, at the back-end of the array.

  As encryption is on a per-drive basis, the computes required are included in the drive enclosure, allowing for a scalable solution, adding encryption with every unit.

  Challenges

  The following challenges should be considered:

  - The cost to the functionality is added with every unit. So, while performance scales, so can costs.
  - Customers might be unable to verify that encryption is enabled and functioning on the array because data is always plain text when it is external to the disk drive.
  - Any approach to encryption at this level also requires interoperability of the encryption implementation across drive vendors to maintain flexibility and customer choice.
  - Bulk drive encryption would not provide key granularity at the LUN/device level, which in many cases would eliminate the possibility of erasing specific confidential projects through key deletion.
  - As driven by the Trusted Computing Group, encryption at this level may follow a different path for validation, an alternative to FIPS 140 yet to be developed. Without a standard to evaluate, it is impossible to understand the disk drive encryption validation proposal.

EMC solutions

- **Disk drive**
  - An alternative to the encryption option for the protection against media theft is EMC Certified Data Erasure. This addresses the same primary use case: protection of disk media containing sensitive data. EMC Certified Data Erasure is available today, as erasure services (for removed drives) and software (for in-frame erasure). EraSure overwrites data multiple times, in accordance with the Department of Defense specification 5220.22-M, removing the data from the media.
  - One consideration is that a minority of drives are not erasable for mechanical reasons. However, customers can keep these drives in a secure onsite location through the EMC Disk Retention Service.

◆ Controller encryption

Another approach might be to implement encryption in the I/O **controller** connected to the disk drives. Some points to consider for this potential implementation are:

- The cost model would be based on a single controller versus multiple drives connected to a single controller.
- The controller approach has the ability to perform encryption at the I/O level, allowing the granularity for key management to be at the LUN or disk level. This approach allows for future support of LUN-based erasure and logical data management.
- The controller approach is drive-independent, not relying on any specific vendor or interface, allowing for all standard tools and failure analysis to be performed.
- In supporting encryption at the controller level, the crypto boundary can be well defined, allowing for FIPS 140-2 validation.
- Encryption and key control is separate from the disk drive containing the encrypted data. This allows the customer to validate the encryption functionality is working and not be concerned with keys leaving with a removed disk drive.

Challenges

- Encryption is on the interface level and is required to support full wire speed versus interface speed in the drive approach.

Tape-level encryption

As part of normal operations, data is frequently written from storage devices to tape for backup data protection or third-party use. Data on tape cartridges becomes susceptible to theft or loss due to the size of the tape cartridge and quantity of the number of tapes to track during normal backup operations. To best protect the data on tape against unintended or unauthorized viewing, it can be encrypted.

There are several approaches to encrypting tape as part of the backup operation:

- Reading encrypted data from application/disk and writing as encrypted data to tape.
- Reading unencrypted data from application/disk and encrypting as part of the backup application.
- Encrypting any or all data sent to tape through an encryption appliance in the network.
- Encrypting any and all data written to the tape through an encrypting tape library or tape device.

Challenges

Tape encryption also presents key management challenges, including:

- Tapes may be stored for an extended period of time before an attempt is made to recover information.
- During the normal process of managing encrypted data, the application may have re-keyed the data on disk, updating all data on the disk to use a new key. This process would present the application with active data using one key and data on tape using an older key. The application must be therefore be able to manage keys for the lifetime of the data, regardless of where the data is stored.

Other resources

Additional information on storage security can be found on EMCs EMC Online Support (https://support.emc.com), including a white paper used for this section, *Approaches for Encryption of Data-at-Rest in the Enterprise - A Detailed Review*. Select best practices for implementing various secure SAN mechanisms are contained throughout this chapter. Some vendor-specific best practices are included in the Chapter 3, "Security Implementation Examples."

Building secure SANs

Many factors need to be considered when building secure SANs. Network and security requirements are often unique to each business environment. The advice of professional security consultants is often sought to balance business security, information accessibility, and performance needs.

Design considerations

Before designing a secure fabric, you need to consider current and pending regulations, management costs, data availability, and network maintenance.

The availability of data is arguably the most important aspect of good SAN design. Prior to the application of security mechanisms, whether they are initiated from customer security policy or IT initiatives, the existing SAN needs to be in a state of stability. The complexity of adding layers of security policy in different zones, from the core to the perimeter of the data center, requires a stable SAN in order to troubleshoot any issues when the security variables are introduced.

Assuring availability can come in many forms. For example, knowing who has physical and administrative access to all components of a SAN, in a well-documented format, is a simple best practice. One main reason for costly downtime is due to physical, unintentional breaches in the environment and not knowing who owns administrative access to the affected components.

When designing the security aspects of the SAN, administrators need to be aware of business availability requirements to avoid "over-securing," which can be costly. SAN scaling and general maintenance impact on security features need to be considered to prevent loss of availability or security features. Through proper redundancy and failover practices, secure availability can almost be guaranteed.

Change control and maintenance of records and procedures have become popular security practices within IT organizations. Change control practices must assure that proposed changes to an environment are documented from start to finish with appropriate approval processes in place throughout the IT organization.

Contingency plans need to be documented as well. All personnel who have access to the environment must be made aware of changes.

Natural disasters are also to be considered when planning secure storage environments. Typically, offsite redundancy and replication is preferred to localization of resources for disaster recovery. Suggestions to maintain secure availability in disaster scenarios range from locating resources to create a sense of obscurity to frequently fire-drilling such disasters that include the loss of strategic key management resources.

2

Implementing RSA Key Manager (RKM) HA Functionality

This chapter provides information on implementing RSA Key Manager (RKM) HA functionality.

- RSA Key Manager (RKM) overview 56
- Configuring the monitor 58

RSA Key Manager (RKM) overview

The RSA Key Manager (RKM) appliances support High availability by clustering two appliances together for failover functionality.

In order to implement seamless failover, RSA recommends using some kind of frontend IP load balancer device or application. The clients will communicate with the RKM cluster appliances using the frontend IP address provided by this IP load-balancer device or software. This frontend device or software will ensure that there is no disruption noticeable to the client in the event of a backend RKM server failure or failover.

IMPORTANT

In the event of a failover, both RKM servers will restart the cleartrust and apache services. Based on the timeout implementation on the clients, this could cause a temporary disruption in client access to the key manager.

The F5 Big-IP Local Traffic Manager (LTM) is an example of an IP load balancer that can be deployed as the frontend for the RKM clusters. Always refer to the *EMC Support Matrix* (ESM) for the most up-to-date information about which IP load balancers are currently supported with the RKM.

The RKM appliance version 2.5.0.3/2.6 and above provide a monitoring tool that integrates with the monitoring feature of the BIG-IP LTM. The following topology, shown in Figure 10 on page 57, shows an example of deployment of the RKM cluster with F5 BIG-IP LTM.

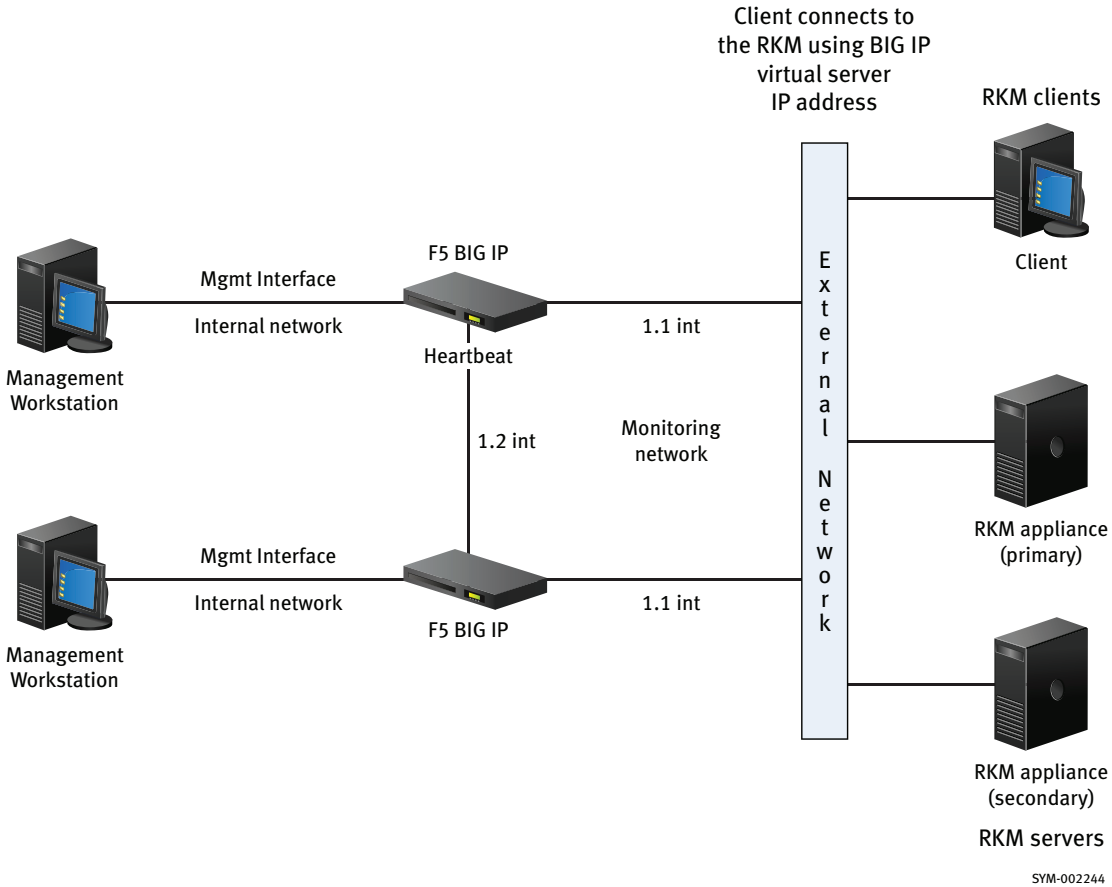


Figure 10 Deployment of the RKM cluster with F5 BIG-IP LTM topology example

As shown in Figure 10, the two BIG-IP LTM appliances are clustered together for redundancy. Each appliance is configured with VLANs to separate the monitoring traffic from the management and heartbeat traffic. The BIG-IP cluster manages backend devices using the concept of pools. The BIG-IP monitors the members of the pool using the customized monitor/s that the user configures. The Virtual server defined on the BIG-IP determines the frontend IP address and hostname that will be used to connect to the clients in the backend.

Configuring the monitor

The steps in this section define the procedure to configure the monitor on the RKM and the BIG-IP LTM appliance. This procedure assumes that the user has already configured the following:

- Two RKM appliances in clustered mode.
- BIG-IP appliance cluster and heartbeat configuration.
- Physical connectivity to the backend RKM servers.
- VLAN settings and interfaces settings on the BIG-IP appliance.
- Node settings with info about the two RKM appliances.

To configure the monitor on the RKM and the BIG-IP LTM appliance, complete the following steps. Each step is further described in this section:

1. Configure the F5 appliance with information about the backend servers.
2. Configure the F5 appliance with the information about the frontend virtual IP address.
3. Configure the health check monitor on the RKM appliances.
4. Configure the F5 appliance to monitor the backend servers using the health check monitor tool.
5. Ensure that the monitoring works as expected.

These steps to configure the health monitor on the RKM and BIG-IP LTM appliance are discussed next in more detail.

1. Configure the F5 appliance with information about the backend servers.
   a. Log into the F5 appliance.
   b. Configure nodes on the F5 appliance with the IP address of the backend RKM servers.
   c. Configure a pool that contains these nodes with the following settings:
      – Leave health monitor setting blank for now.
      – Load balancing method = round robin.
      – Priority Group activation = Less than 1 member (given that there are two nodes in the pool).

– Add the two RKM appliances with the higher priority # to the active server. In the event of an RKM failover and failback, the user should change the priority of the RKM servers prior to manually bringing up the connection from the F5 server to previously failed appliance. Step 4 further details the manual resume setting.

2. Configure the F5 appliance with the information about the frontend virtual IP address.

   Configure the virtual server information on the BIG-IP appliance using the IP address that the frontend clients would use to connect to the backend servers. In the event of a failure of one of the backend RKM servers, the virtual server IP address will automatically redirect client traffic to the active backend server. This will ensure seamless operation of client operations on the frontend.Consult the *BIG-IP Configuration Guide*, located at http://www.f5.com, for details.

3. Configure the health check monitor on the RKM appliances.

   The health check monitor, using a URL via external HTTPS, allows a load-balancer (or other external monitoring system) to monitor and verify whether or not an RKM appliance is working properly and can receive RKM client traffic. Complete the following steps to use the health check monitor:

   a. Create a client certificate with password as “Password1” using tools like openssl.

      Note: The password for the client certificate must be “Password1”. The health check monitor assumes that you are using a dummy client certificate.

   b. Log in to the primary RKM frontend GUI as kmsadmin (https://<rkm server hostname>/KMS).

   c. Create an identity in the key management solution server using the client certificate created in Step 1.

   d. Create a Key Class inthe key management solution server.

   e. Generate a key in the Key Class created in Step d.

   f. Type the following into the browser to begin configuring your monitor:

      https://<rkm server hostname>/rkmawa/healthCheck.do

g. Append the information in the browser with the following parameters and include values for each:

- **keyclass** – The key class created in the key management solution Admin UI.
- **rootca** – Root certificate file that corresponds to the identity associated with a key class.
- **client** – Client p12 file that corresponds to the identity associated with a key class.

  Note: In case any of the parameter values has a white space, provide the value surrounded by single quotes.

  The syntax is
  ?keyclass='[value]'&rootca='[value]'&client='[value]'

  The entire URL in the browser should look similar to the following:

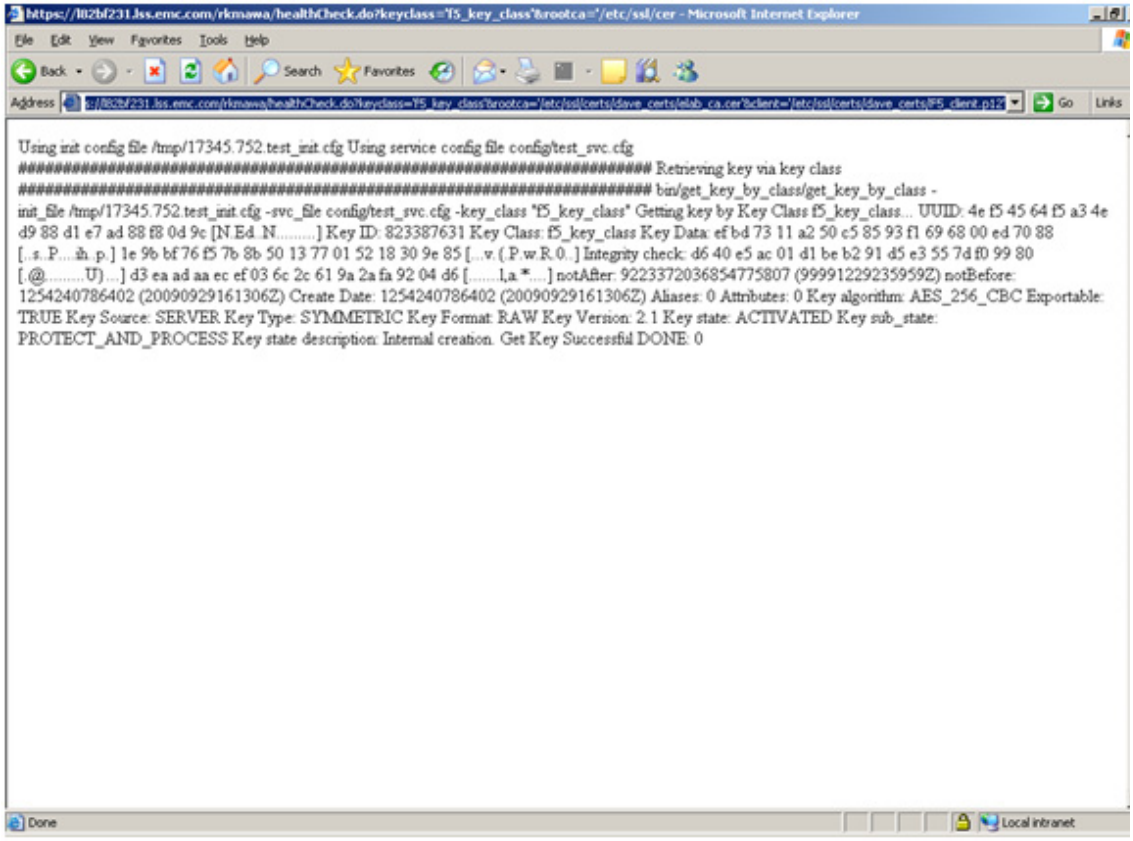
```
https://RKMAPPLIANCE.RKM.COM/rkmawa/healthCheck.do?keyclass='Key
   Class'&rootca=/demoCert/cacert.pem&client=/demoCert/client.p12
```

h. Press **Enter** or click **Go**.

i. Accept the certificate, if presented with one.

j. Log in to the cleartrust (this may be required the first time).

The page displayed will be similar to the following:


https://l02bf231.lss.emc.com/rkmawa/healthCheck.do?keyclass=f5_key_class&rootca=/etc/ssl/cer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address
Go Links
Using init config file /tmp/17345.752.test_init.cfg Using service config file config/test_svc.cfg
Retrieving key via key class
bin/get_key_by_class/get_key_by_class -
init_file /tmp/17345.752.test_init.cfg -svc_file config/test_svc.cfg -key_class "f5_key_class" Getting key by Key Class f5_key_class... UUID: 4e f5 45 64 f5 a3 4e
d9 88 d1 e7 ad 88 f8 0d 9c [N.Ed..N.....] Key ID: 823387631 Key Class: f5_key_class Key Data: ef bd 73 11 a2 50 c5 85 93 f1 69 68 00 ed 70 88
[..s..P....ih..p.] 1e 9b bf 76 f5 7b 8b 50 13 77 01 52 18 30 9e 85 [...v.{.P.w.R.0..] Integrity check: d6 40 e5 ac 01 d1 be b2 91 d5 e3 55 7d f0 99 80
[.@.....U}....] d3 ea ad aa ec ef 03 6c 2c 61 9a 2a fa 92 04 d6 [.....l,a.*....] notAfter: 9223372036854775807 (99991229235959Z) notBefore:
1254240786402 (200909291613062) Create Date: 1254240786402 (200909291613062) Aliases: 0 Attributes: 0 Key algorithm: AES_256_CBC Exportable:
TRUE Key Source: SERVER Key Type: SYMMETRIC Key Format: RAW Key Version: 2.1 Key state: ACTIVATED Key sub_state:
PROTECT_AND_PROCESS Key state description: Internal creation. Get Key Successful DONE: 0
Done
Local intranet

k. Confirm that the last sentence in the text displays, "Get Key Successful DONE 0."

   This confirms that the health check monitor is set up correctly on the RKM server.

l. Copy the client certificate created into the secondary server in the same location as that on the primary server.

4. Configure the F5 appliance to monitor the backend servers using the health check monitor tool.

   a. Create a new monitor of type = https with the following parameters:

      – Interval = 30 seconds
      – Timeout = 91 seconds

- Manual Resume = Yes

  This implies that in the event of a backend RKM failure the user must log into the F5 to manually bring up the F5 connection to the previously failed RKM server.

- Send String = GET /rkmawa/healthCheck.do? healthCheck.do?keyclass='Key Class'&rootca=/demoCert/cacert.pem&client=/demoCert /client.p12' HTTP/1.1\r\nHost:\r\nConnection: Close\r\n\r\n
- Receive String = Get Key Successful

b. Leave the rest of the configuration as default.

c. Provide the client certificate, if required.

5. Ensure that the monitoring works as expected.

   Confirm that the BIG-IP appliance is able to monitor the RKM appliances using the RKM scripts. This can be checked using the BIG-IP pools information under the **Members** tab. Figure 11 on page 63 shows an example of a working health monitor.

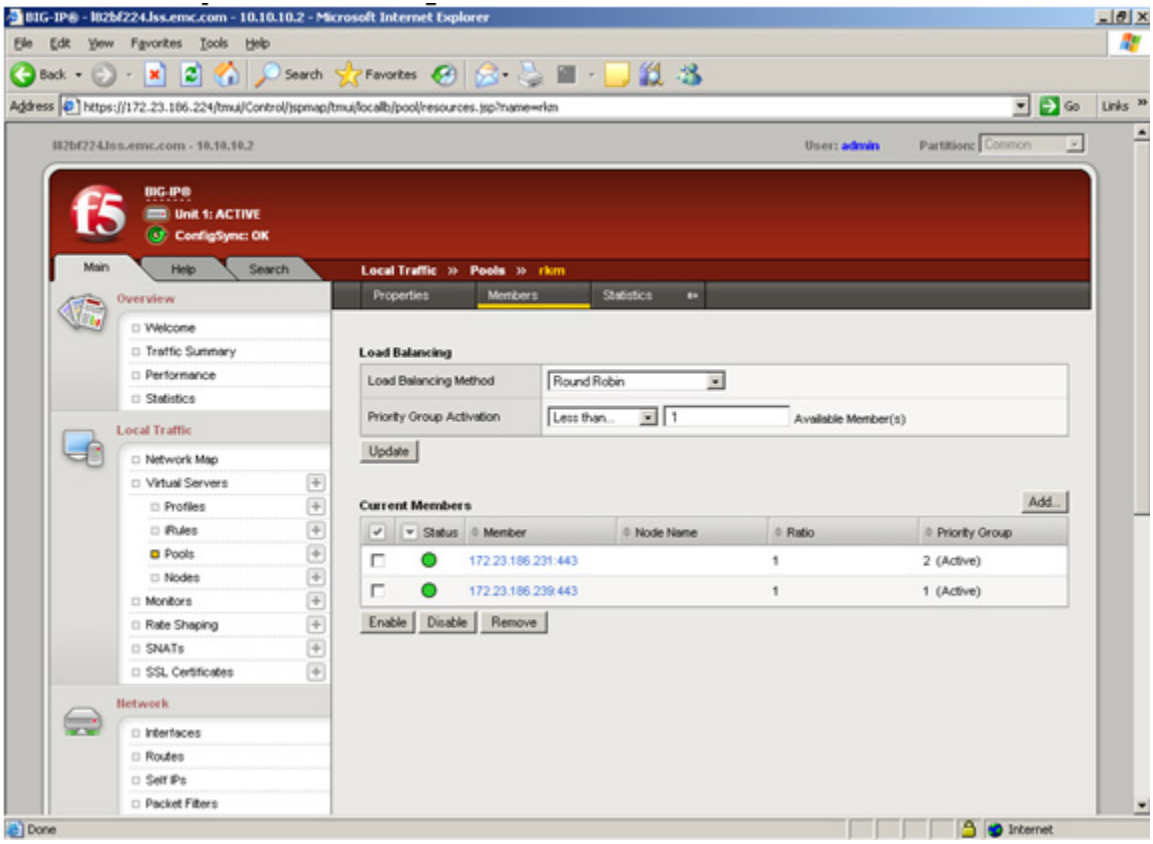


Figure 11 Working health monitor example

3

Security Implementation Examples

The following security implementation examples are discussed in this section, with emphasis on supported features and best practices.

- EMC Celerra iSCSI data security 66
- Brocade 69
- Brocade M Series 76
- Cisco 82

EMC Celerra iSCSI data security

EMC Celerra® iSCSI features can be leveraged to establish a secure environment for iSCSI sessions. Each of these measures provides a level of security when implemented separately. When combined, security is increased immensely. For instance, just implementing LUN masking provides a level of security, but an unscrupulous person could spoof an Initiator Qualified Name (IQN) in order to gain access to a LUN. Implementing CHAP authentication and firewall rules, in addition to LUN masking, severely restricts such attempts.

Supported features

The following are EMC-supported features:

- iSCSI LUN Masking
- CHAP Authentication
- iSCSI Port Protection
- VLAN Tagging
- Filtering by IP address and port

Best practices

The following is a list of best practices to consider:

- Celerra iSCSI security features can be implemented through wizards. LUNs can be masked while allowing for multiple access options for cluster node joint access through an easy-to-use Data Mover graphical interface (Figure 12 on page 67).

  A LUN mask creates an association between the iSCSI target (LUN) and the IQN. The Celerra system will deny access to a LUN unless a mask has been configured to associate the LUN with the IQN.

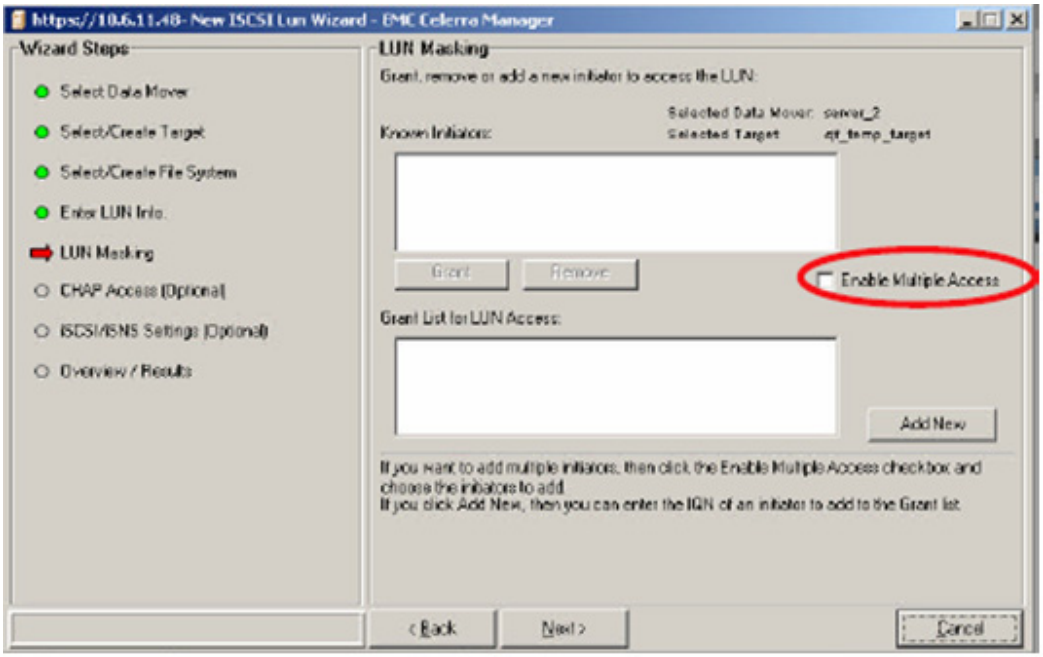

https://10.6.11.48- New ISCSI Lun Wizard - EMC Celerra Manager
Wizard Steps
Select Data Mover
Select/Create Target
Select/Create File System
Enter LUN Info.
LUN Masking
CHAP Access (Optional)
iSCSI/iSNS Settings (Optional)
Overview / Results
LUN Masking
Grant, remove or add a new initiator to access the LUN:
Selected Data Mover: server_2
Known Initiators:
Selected Target: qt_temp_target
Grant
Remove
Enable Multiple Access
Grant List for LUN Access:
Add New
If you want to add multiple initiators, then click the Enable Multiple Access checkbox and choose the initiators to add.
If you click Add New, then you can enter the IQN of an initiator to add to the Grant list.
< Back
Next >
Cancel

Figure 12 LUN masking

- VLAN technology allows for the creation of smaller network domains of trusted systems. Since it may not be desirable to allow all of the systems in the environment to have iSCSI access to the Celerra system, enabling VLAN tagging on the iSCSI interface for Celerra provides access control at the switch to police which systems are able to mount iSCSI devices.
- Implement CHAP authentication on initiators and targets, based on a shared secret, random challenge, and secure one-way hash. Celerra supports CHAP authentication (see Figure 13 on page 68). The default settings do not require CHAP be enabled; however, data access in real time (DART) parameters can be configured to force all initiators to use CHAP to authenticate to the Celerra target. The Celerra system offers several options to help manage secure environments.

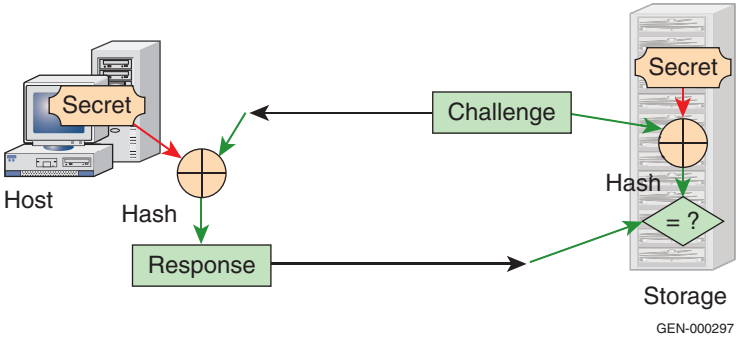


Figure 13 CHAP authentication

References

For information on installing iSCSI host components and best practices, refer to EMC Celerra Network Server documentation, located at EMC Online Support (https://support.emc.com).

Brocade

Brocade provides flexible features to assist the IT professional in safeguarding the SAN. In addition to traditional hardware security features, since Brocades Fabric OS version 5.2, more Secure Fabric OS security features are part of Brocades standard license. In this section, some of these security features are noted with implementation recommendations. Implementation details are provided in the *Fibre Channel SAN Topologies TechBook*, available through the E-Lab Interoperability Navigator, **Topology Resource Center** tab, at http://elabnavigator.EMC.com, or in the referenced Brocade documentation.

Security features

Brocade provides policy-based security protection for more predictable change management, assured configuration integrity, and reduced risk of downtime. Some of these Brocade security features are listed in Table 2 under the categories of Authentication, Authorization, Accountability, Encryption, and Integrity. Arguably, a feature may be listed under one or more of these categories. Product-specific, conditionally and unsupported features are specifically listed in "EMC conditionally or unsupported features" on page 71.

Table 2 Brocade security features (page 1 of 3)

Category	Feature	Description
Authentication	Switch-to-switch	Switch-to-switch (E_Port) authentication using Fibre Channel Certificate Authentication Protocol (FCAP). Brocade provides commands to disable switch-to-switch FCAP authentication and to select an alternate authentication protocol such as DH-CHAP. DH-CHAP is a secrete-based authentication and key management protocol that supports both switch-to-switch and host-to-switch authentication.
	Password administration and account lockout	Features and policies are included in Admin Domain.
	Switch-Host	RADIUS or TACACS+ is supported to ensure that only authorized devices access protected networks.

Table 2 Brocade security features (page 2 of 3)

Category	Feature	Description
Authorization	Zoning	Hardware enforced WWN zoning.
	Port controls	Supports E_Port, L_Port, and G_Port lock down, Port Binding, and persistent port disable controls.
	Insistent Domain ID [FICON]	Allows a switch to insist on a specific Domain ID prior to joining a fabric.
	LSAN technology	Proprietary to Brocade and similar to Cisco VSAN ideology. Allows storage administrators the flexibility to isolate environments logically.
	Role-Based Access Control (RBAC)	ACL support (Switch and Port Binding) policies are identified by the following specific names and cannot co-exist. The policies are case sensitive. The ACL policies can be distributed to databases per-switch or fabric-wide. The two policy types are: • Device Connection Control (DCC) policies – used to restrict which device ports can connect to other switch ports • Switch Connection Control (SCC) policies – used to restrict switches from joining the fabric or joining another switch.
Accountability	SNMP v3 standard for monitoring and managing	SNMP access-control lists prevent/get/set operations to individual host or IP addresses. SNMP agent and Management Information Base (MIB) are located on each switch.
	HTTPS	Can be used with Web Tools for managing the switch.
	SNMP trap configurations	ISL and security thresholds can be set.
	Virtual Fabric with Administrative Domains	Provides for auditing of user and security related events, such as configuration changes, security, zoning events, and firmware download.

Table 2 Brocade security features (page 3 of 3)

Category	Feature	Description
Encryption	SSH (Secure shell access) for ensuring network security encrypted sessions	Uses the SSH daemon, server-side initiated certificate. Nothing is needed on the switch.
	SSL (Secure Socket Layer) supporting SSLv3 (128-bit encryption) for support of HTTPS.	Certificates must be generated and installed on each switch to enable SSL.
	IPSec Encryption	Services for the PB-48000B-18i line card (FR4-18i) and 7500.
Integrity	MD5 & SHA-1 hashes	DH-CHAP supports MD-5 and SHA-1 algorithm-based authentication.
	SFC (Secure File Copy)	SFC of configuration uploads and downloads.

EMC conditionally or unsupported features

For the most up-to-date listing of EMC conditionally or unsupported product features, please refer to the current product release notes available on EMC Online Support (https://support.emc.com).

- PKI (Public Key Infrastructure) — The Fabric OS 5.2.0 SFOS migration strategy of removing dependencies on a Brocade PKI is based on the assumption that there is limited demand for PKI-based solutions and that administrators will use FC-SP standards for switch to switch authentication. Brocade no longer includes a PKI Certificate as part of the installed Secure Fabric OS. If you wish to activate Secure Fabric OS on a supported director or switch, you must contact Brocade to obtain a PKI certificate.
- Secure FOS is not supported by EMC. FR4-18i (ED-48000B) blades can be used in a switch running Secure FOS. However, adding a reference to one of the ports added by PB-48K-48 (known as ports 256-383) in a DCC policy can only be performed on a FOS 5.2 or newer switch since older versions of FOS limit the port numbers to a maximum of 255. Because of this limitation, the FCS switches must be running FOS 5.2 or later firmware. The FCS switch is the only switch that can be used to modify security policies.
- Fast Write and Tape Pipelining are not supported in conjunction with secure tunnels.
- Jumbo frames are not supported on secure tunnels.

- ICMP redirect is not supported for IPSec-enabled tunnels.
- Only a single secure tunnel is allowed on a port. Non-secure tunnels are not allowed on the same port as secure tunnels. Modifying operations are not allowed on secure tunnels. To change secure tunnel configurations, you must first delete the tunnel and then create a new one with the desires options. Only a single route is supported on an interface with a secure tunnel.
- An IPSec tunnel cannot be created using the same local IP address if ipperf is active and using the same local IP address (source IP address). Unidirectional supported throughput is ~104Mbytes/s and bidirectional supported throughput is ~90Mbytes/sec. An IPSec tunnel takes longer to come online than a non-IPSec tunnel. The user is not informed with the IPSec mismatch RAS event when configuring a tunnel with IPSec mismatch on either ends.

Best practices

The best practice recommendations described in this section were developed based on EMC's current understanding of the general security environment and the capabilities of EMC's product(s). Security is also impacted by your requirements and your IT environment along with newly discovered vulnerabilities, security threats, and technologies identified on an almost daily basis. Changes to any of these factors may impact the applicability and effectiveness of these best practice recommendations.

Implement Role Based Access Control (RBAC)

Fabric OS 5.0.1 has three different roles: User, Admin, and Switch Admin. Another four roles (Operator, Fabric Admin, Zone Manager, and Basic Admin) have been added as of Fabric OS 5.2. Table 3 on page 73 shows the functional area and capabilities for each of these roles. Refer to the *Brocade Administrator's Guide* for implementation details.

Table 3 Implement Role Based Access Control (RBAC)

Functional area	User	Switch Admin	Admin	Operator	Zone Manager	Fabric Admin	Basic Admin
Zone Configuration	V	V	M	V	M	M	V
Environmentals	V	M	M	M	V	M	P
Enable/Disable Ports	V	M	M	M	V	M	P
Logs (RAS)	V	M	M	M	V	M	P
Security Settings	V	V	M	V	V	M	V
Switch Configuration	V	M	M	V	V	M	P
Switch Management	V	M	M	M	V	M	P
Port Configuration	V	M	M	V	V	M	P
SNMP	V	M	M	V	V	M	P
Diagnostics	V	M	M	M	V	M	P
Devices	V	M	M	V	V	M	P
User Management	X	X	M	X	X	X	X
Fabric Watch	V	M	M	M	V	M	P
APM	V	M	M	V	V	M	V
Admin Domain Mgmt.	X	X	M	X	X	X	X

Legend:

V = View Only
M = View and Modify
P = Partial
X = N/A

Implement auditing

Use capabilities and procedures to capture, record, and track significant events. Captured event information should account for:

- Which users are making changes from an external source (username).
- Where they are logged in from (IP address).
- Type of management interface (Telnet, Web Tools, etc.).

Since the types of events being audited may be somewhat frequent, the handling of these events can be streamed off the switch to a remote host running a SYSLOG server. Scheduled at a logical and regular interval (such as every day at midnight) an auditor could review events to look for activity such as configuration changes, firmware downloads, etc., that may be useful for network troubleshooting or security breaches. Refer to Brocade documentation for implementation details.

Implement hardware-enforced WWPN zoning

Securing ports can be a hindrance to flexibility of managing switches but is typical of securing an environment. Trade-offs should be expected. Recommendation is to implement hardware-enforced WWPN zoning. No license is required and this feature is enabled by default. Refer to Brocade documentation for implementation details.

Implement port controls

Use port controls such as E_Port lockout, L_Port lockdown, G_Port lockdown, and persistent port disable. These features help to prevent physically connected nodes from logging into an unintended or restricted environment. By manually setting the port configuration, you bypass the ports default auto-configuration. Refer to the *Brocade Fabric Manager's Administrator's Guide* for switch dependent implementation details.

Secure CLI sessions

Minimally, password protect service port and IP connections. Additional uses of switch-to-switch and switch-to-host authentication mechanisms are suggested. Refer to Brocade documentation for implementation details.

Related Brocade documentation

The following documentation is available on Brocade's website at http://www.Brocade.com:

- *Brocade Fabric OS Administrator's Guide*; Publication Number:53-1000043-02
- *Brocade Fabric Manager's Administrator's Guide*; Publication Number 53-1000042-01
- *Brocade Fabric Manager's Administrator's Guide*; Publication Number 53-1000043-02

- *Secure Fabric OS Administrator's Guide*, Chapter 2, "Adding Secure Fabric OS to the Fabric," for a description of how to obtain certificates from the Brocade Certificate Authority.

Brocade M Series

Brocade McDATA (hereafter referred to as Brocade M) switches, directors, and software management applications offer components of Brocade's SANtegrity Security Suite.

SANtegrity ensures that Fibre Channel traffic is not redirected by unauthorized access. SANtegrity supports hardware-enforced zone parameters to protect from DoS attacks. Both zone member specification by port or WWN and software-enforced Fabric Name Server zoning is supported. In mainframe FICON environments, FICON cascading is enabled when SANtegrity is installed.

The SANtegrity Security Suite includes both standard and enhanced licensed features.

Standard features include:

- Advanced Zoning

  Advanced zoning enforces zone parameters at the ingress port to protect from DoS attacks on the Fabric.

- Role Based Accounting Control (RBAC)

  RBAC provides management zoning.

- Secure Access

  Provides locked down interface management and improved performance.

- Secure Platform

  Provides secure techniques for communication with storage network devices and for secure client server communication.

Enhanced licensed features include:

- Binding

  Binding is a set of features that locks a fabric into an intended configuration.

- Authentication for both FC and IP block-based protocols

  SANtegrity Authentication is based upon the interoperable authentication protocol, DH Challenge Handshake Authentication Protocol (CHAP), recommended as the Fibre Channel standard for security (FC-SP) in storage networks. Each

device participating in a storage network proves its identity through the supported FC-SP, iSCSI, FC-GS, FC-SB, and iFCP protocols.

- Security Center management for Brocade's EFCM and SANavigator products.

  SANtegrity Security Center manages the administration of device settings, generates logs and reports based on security policies, and promotes both monitoring and planning.

A license key specifies the maximum number of switch ports, the number of clients you can run, the expiration date of a temporary license, and the licensed optional features.

Security features

Brocade M features are noted in Table 4 under the categories of Authentication, Authorization, Accountability, Encryption and Integrity. Arguably, a feature may be listed under one or more of these categories.

Table 4 Brocade M security features (page 1 of 2)

Category	Features	Description
Authentication	Users	Adding, deleting, editing, defining roles, and password maintenance for authorized users.
	Software	Configure authentication settings for management access of both in-band and out-of-band software access.
	Devices	Define CHAP Secrets, Port Authentication Sequences (i.e., RADIUS, then Local: Radius Only; Local Only), and adding/configuring/removing authenticated devices.
	Ports	Port Authentication allows user to override authentication settings of specific ports.

Table 4 Brocade M security features (page 2 of 2)

Category	Features	Description
Authorization	Zoning	Hardware-enforced WWN zoning is enabled by default and does not require configuration. Zoning is enforced in the ASIC route tables at the ingress port. When Class 3 frames are sent to none zone members, they are dropped.
	Port Binding	Port Binding has access control policy by port. You must configure F_Ports or E_Ports WWNs on an individual port basis.
	Switch Binding	Permits specific F_Ports or E_Ports to connect to a particular switch. The feature may be enabled by activating the "Enterprise Fabric" mode or by activating it directly.
	Fabric Binding	SANtegrity licensed option that permits specific switches to join a fabric. Binding has an access control policy by switch. The Insistent Domain ID feature allows the switch Domain ID to be statically set. **Note:** This feature MUST be enabled with Fabric Binding.
	Port Controls	Standard port type controls as well as persistent port blocking..
Accountability	Accounting Services	Log information for each management session in a switch. Can be implemented locally or remotely (RADIUS). Syslog available for centralized repository of audit logging; different logs can be configured through E/OS.
	Logs	Email notifications are sent and switch logs are updated when devices attempt to log in to a port with Port Binding.
	Role-Based Access Control	Access via CLI or SNMPv3. Secure management zone and reporting.
	Password Management	Password expiration through CLI and Connectrix Manager. To eliminate the possibility of password "guessing", as of E/OS 9.1 switches have the feature set of configuring and retaining a history of up to three passwords. Administrators who require users to change passwords periodically can do so with this feature.
Encryption	SSH/SSL	SSH and SSL for ensuring network secure encrypted sessions. SSH through CLI since E/OS 7.1. SSH for Connectrix Manager (EFCM) between GUI and switches since E/OS 8.1. SNMPv3 for MIB management.
Integrity		Supports SSH v2, SNMPv3, SFTP, AES, MD5 and SHA 1

Best practices

The best practice recommendations described in this section were developed based on EMC's current understanding of the general security environment and the capabilities of EMC's product(s). Security is also impacted by your requirements, your IT environment, and by newly discovered vulnerabilities and technologies. Changes to any of these factors may impact the applicability and effectiveness of these best practice recommendations.

Common recommendations include changing default passwords, using port controls to disable unused ports or management interfaces, using SSL or SSH, and limiting physical access to FC switches. The following are additional best practices when configuring Brocade M secure fabrics.

- "Hardware enforced WWPN zoning" on page 79
- "Fabric Binding" on page 80
- "Switch Binding" on page 80
- "Port Binding" on page 80
- "EFCM reporting and auditing" on page 81

Hardware enforced WWPN zoning

Zoning in Brocade M devices is similar to Brocade and Cisco devices in that zoning minimizes fabric disruptions. The security administrator should take advantage of zoning's ability to increase security measures by preventing data loss, corruption, or attacks by specifying which devices can access one another. Brocade M switches allow for this by implementing hardware-enforced WWPN zoning. No license is required for this feature and it is enabled by default.

Brocade M recommends creating logical subsets of zones/user groups. Authorize access rights to those specific zones for specific user groups so that unintentional usage will not occur and the protection of confidential data from unauthorized access is consistent. Creating groups of devices that are separate from devices in the rest of the fabric, and zoning them, will allow processes to be performed on devices in one group without interrupting the other. For example, use single initiator zoning and keep your backup traffic separate from primary traffic.

Zoning practices and the Safe zoning mode (introduced in EOS 9.01.00) prevents switch merges from inadvertently creating

unintended zone sets and prevents default zones from being enabled. It also prevents merges of fabrics with zone sets that do not match.

After the environment has been configured and all hosts see the appropriate volumes, configure the following Enterprise Fabric mode security features (each discussed in more detail in this section):

- Fabric Binding
- Switch Binding
- Port Binding

Fabric Binding

The Brocade M Fabric Binding feature provides security from intentional and accidental fabric merges. Fabric Binding permits only specified switches to join a fabric. It is set on a fabric-wide basis, not on an individual switch basis. World Wide Names (WWNs) of the switches are used to determine which switches may participate in a fabric. In McDATA Enterprise Fabric Mode, Fabric Binding cannot be disabled. Configuration details and information on how to modify the Fabric Binding Membership List (FBML) can be found in the "Configuring Security" chapter in the *EFCM Basic User Manual*, located at http://www.Brocade.com.

Switch Binding

Switch Binding permits specific F_Port or E_Port connections to a particular *switch*. The feature may be enabled by activating the Enterprise Fabric mode or by activating it directly. WWNs of a node or switch are used to determine which nodes or switch may connect to a switch. Switch Binding is configured on an individual switch basis. Nodes already participating in the fabric when Switch Binding is activated will automatically be included in the Switch Binding Membership List (SBML). In Switch Binding, a node in the SBML may connect to any port on the switch.

Port Binding

Port Binding permits specific F_Ports or E_Ports to connect only to a particular *port* of a switch.

Port Binding is more granular then Switch Binding. WWNs of F_Ports or E_Ports must be configured on an individual port basis. In Port Binding, a WWN or an Alias of a node is assigned to a port and it may only connect to that port. Configuration details can be found in the "Configuring Security" chapter in the *EFCM Basic User Manual*, located at http://www.Brocade.com.

EFCM reporting and auditing

Use the reporting and auditing tools in Connectrix Manager to track logins, operating parameters, and zoning changes. The following events are logged and may be used to satisfy auditing requirements. Implementation details can be found in the referenced *EMC Connectrix Manager User Guide* located at EMC Online Support (https://support.emc.com).

- Defining a new product
- Modifying product attributes
- Deleting product definitions
- Defining a new user
- Modifying user administration
- Deleting user administration
- Creating a zone set
- Modifying SNMP options

Cisco

Cisco provides a comprehensive security framework within SAN-OS. Licensing is required for some enhanced security features including FC-SP authentication, port security, LUN zoning, IPSec, and VSAN-based access control. Licensing and implementation details can be found in the referenced *Cisco MDS 9000 Family Configuration Guide* and *Cisco MDS 9000 Family Fabric Manager User Guide*, both available at http://www.cisco.com.

Security features

Cisco features are listed in Table 5 under the categories of Authentication, Authorization, Accountability, Encryption, and Integrity. Arguably, a feature may be listed under one or more of these categories. Product-specific conditionally and unsupported features are listed following Table 5.

Table 5 Cisco security features (page 1 of 2)

Category	Feature	Description
Authentication	Switch-to-Switch	FC-SP for Switch-to-Switch and Host-to-Switch Authentication using DH-CHAP. Supports MD5 and SHA-1 algorithm based authentication. Configuring the DH-CHAP feature requires the ENTERPRISE_PKG license.
	Host-to-Switch	RADIUS or TACACS+ is supported to ensure that only authorized devices access protected networks.
Authorization	Zoning	Supports N_Port, Fx_Port, iSCSI, LUN, Read-Only, and Broadcast Zones, as well as, hardware enforced WWN zoning.
	VSAN technology	Proprietary to Cisco and similar to Brocade LSAN ideology that allows storage administrators the flexibility to isolate environments logically. Port Security is activated on a per VSAN basis.
	Switch Access Control	Supports SSH v2, SNMP v3, and Role Based ACLs.
	Persistent Port Disable	A disabled port remains disabled through offline/online changes and reboots.
	Fabric Binding	FICON Pre-3.x; VSAN Post 3.x extends port security to enable ISLs only between specified switches.

Table 5 Cisco security features (page 2 of 2)

Category	Feature	Description
Accountability	Accounting Services	Log information for each management session in a switch can be implemented locally or remotely (RADIUS).
	Port Tracking	Monitors and detects failures that cause topology changes.
	Role-Based Access Control	RBAC via RADIUS/TACACS+ features pre-defined roles and customization ability to limit administrators and users on a VSAN basis. For management access via CLI or SNMPv3, the same usernames and passwords are used to gain access to the switch.
Encryption	Digital Certificates	Management tools utilize SNMPv3 to communicate between switch and GUI.
	SSH	SSHv2 encrypts and authenticates traffic between switches and management stations instead of Telnet. Host keys RSA, RSA1, DSA, and AES are available.
	IPSec	Encryption services for the MPS 14/2 or MDS 9216i.
Integrity	Protocol Methods	Supports SSH v2, SNMPv3, SFTP, AES, MD5, and SHA 1
	IPSec	FCIP and iSCSI connections use IKEv1 and IKEv2 protocols. Protects and authenticates IP packets between participating devices. Enterprise Package required.

Conditionally or unsupported features

For a listing of EMC conditionally or unsupported product features, please refer to the most current version of the *Connectrix MDS 9000 SAN-OS Release Notes* residing on EMC Online Support (https://support.emc.com).

- Cisco implementations of LUN zoning are unsupported
- Read-only zones are unsupported
- Cisco implementations of IPsec and AES Encryption for iSCSI are unsupported

Best practices

The following best practices are recommended.

Implement zoning and VSANs

Zoning and VSANs are Cisco's natural security features that can be applied to isolate traffic within the fabric. By segregating groups of hosts and storage you are not protecting the environment via authentication or data encryption at this layer, but you are providing availability and fault tolerance. See Cisco documentation for implementation details.

VSANs should be created where applicable and where management of the VSANs will not over-complicate the environment. Ports available for VSANs are 1-4094. By default, all ports fall under VSAN port 1. To avoid possible WWN spoofing, place all unused ports in backup VSAN 4094 and then build separate VSANs in 2-4093.

It is recommended to use static domains to avoid any possible merge conflicts and manageability conflicts. Inter-VSAN routing should be used sparingly for the resources that need to be shared.

Implement hardware-enforced WWPN zoning

Securing ports can hinder the flexibility of managing switches, but are typical of securing an environment. Tradeoffs should be expected. Recommendation is to implement hardware-enforced WWPN zoning. Hardware-enforced zoning has a level of fabric enforcement not provided by software-enforced zoning. Hardware-enforced zoning is applied to every FC frame at the switch port.

Implement port controls and security

Use port controls to eliminate the dangers of having users, intentionally or not, misuse a port that has the default 'auto' mode port settings. To avoid this danger, configure 'port' mode on all switch ports, shut down all used ports, and only allow connections from expected device types by specifying N_Port, E_Port, F_Port, and FL_Port settings.

Port security prevents unauthorized access to a switch port by binding specific WWN access to one ore more given switch ports. Complimentary to port security is WWN-based zoning which zones the switching logic to frames based on the WWN, and not the physical port, on a device. With this logical security, spoofing can be a problem.

Use port security features to:

- Bind devices to switch
- Bind devices to a port
- Bind to a group of ports in case of individual port failure
- Bind switches at ISL ports; not just individual switch

When enabling Port Binding, consider the impact of choosing whether or not device-to-switch or switch-to-switch port security is enabled and assure that it does not impact something that should be accessing a specific port. When these features are enabled, it will reject login requests from unauthorized FC devices as well as report attempts to the SAN administrator. Refer to Cisco documentation for implementation details.

Implement Fabric Binding

Fabric Binding allows access to fabric devices based on identity attributes. All Cisco MDS 9000 switches enable fabric-wide authentication from one switch to another or from a switch to a device.

You can perform switch and host authentication locally or remotely through the DH-CHAP protocol. To configure the Enterprise Package licensed DH-CHAP authentication feature using the local password database, please refer to the *Cisco MDS 9000 Family Configuration Guide*. DH-CHAP based authentication should be employed to prevent spoofing or hijacked WWNs where port security is vulnerable.

RADIUS / TACAC+ provides for centralized management of access control when multiple switches are used. Please note that for switch-to-switch authentication, relying only on remote authentication creates a single point of failure.

Enable port tracking

The port tracking feature monitors and detects failures that cause topology changes or bring down links connecting devices. When enabled and explicitly configured, the Cisco SAN-OS software monitors the tracked ports and alters the operational state of the linked ports upon detecting a link state change.

Secure Management Access

Securing Management Access can be done at all points of the data path. Cisco supports SNMPv3, SSH, and SSL. It is recommended that Telnet, SNMPv2, and HTTP be disabled. Use of VPNs (Virtual Private Networks) are recommended for remote management of an environment.

Another Cisco features known as a VLAN (IP Networks) is recommended to isolate management traffic that commonly connects to the SAN management stations, like Cisco Fabric Manager.

Implement Role Based Access Control accounting services

SAN-OS provides Role Based Access Control (RBAC) for management access of the Command Line Interface (CLI) and Simple Network Management Protocol (SNMP). In addition to predefined roles, up to sixty four (64) roles can be defined. The roles describe the access control policies for various feature-specific commands on one or more VSANs. CLI and SNMP share the same usernames and passwords (i.e., a single administrative account for each user) to gain access to the switch. Predefined roles include:

- Network Operator (read only access)
  - Show Commands
  - Exec mode file system commands
  - Basic network diagnostics
- Network-admin (read/write)
  - Access to all CLI commands

Best practices are to use roles to give access to groups of commands and to deploy them on a per-VSAN basis and by administrative function to avoid over-managing the environment. Roles can have access permissions assigned to them either as CLI or SNMP users. See Cisco documentation for details on how to implement RBAC.

Related Cisco documentation

For Cisco documentation, please refer to http://www.cisco.com.

- *Cisco MDS CLI Configuration Guide; Release 3.x*
- *Cisco 9000 MDS Family Configuration Guide, chapter on Configuring IPSec Network Security*
- *Cisco MDS Storage Networking Fundamentals; Version 2.1 Cisco / Firefly Communications*

- *MDS 3.0 Hardware Software Overview; March 2006 Customer Assurance Engineering*
- *EMC Connectrix MDS 9000 SAN-OS Release Notes*, available on EMC Online Support (https://support.emc.com)

4

Cisco MDS 9000 Family Storage Media Encryption (SME)

Cisco SME (SME) is a standards-based encryption solution for heterogeneous tape libraries and virtual tape libraries. This chapter discussing the following topics:

- Overview 90
- Key features 91
- Terminology 93
- Topology guidelines 94
- Single fabric SME cluster deployment 99
- Key hierarchy overview 103
- Implementation best practices 110

Overview

Cisco MDS 9000 Family Storage Media Encryption (Cisco SME) for the Cisco MDS 9000 family switches offers an enterprise level, scalable SAN security solution. Cisco SME enables encryption capability in an existing fabric service for Fibre Channel SANs with minimal reconfiguration effort. The following section provides a brief overview about the key hierarchy, key management, best practices and basic features supported for SME. Please refer to the Cisco's configuration guide and release notes for best practices and restrictions.

Cisco SME (SME) is a standards-based encryption solution for heterogeneous tape libraries and virtual tape libraries. Cisco SME is managed with Cisco Fabric Manager and a command-line interface (CLI) for unified SAN management and security provisioning.

Key features

Key features of storage media encryption (SME) include:

- Encrypts data stored on tape(s) to protect against tape loss/theft
- Provides comprehensive key management
- Is Regulatory compliant
- Uses FIPS Level-3 System Architecture
- Provides transparent fabric service

Figure 14 shows an example of Cisco SME.

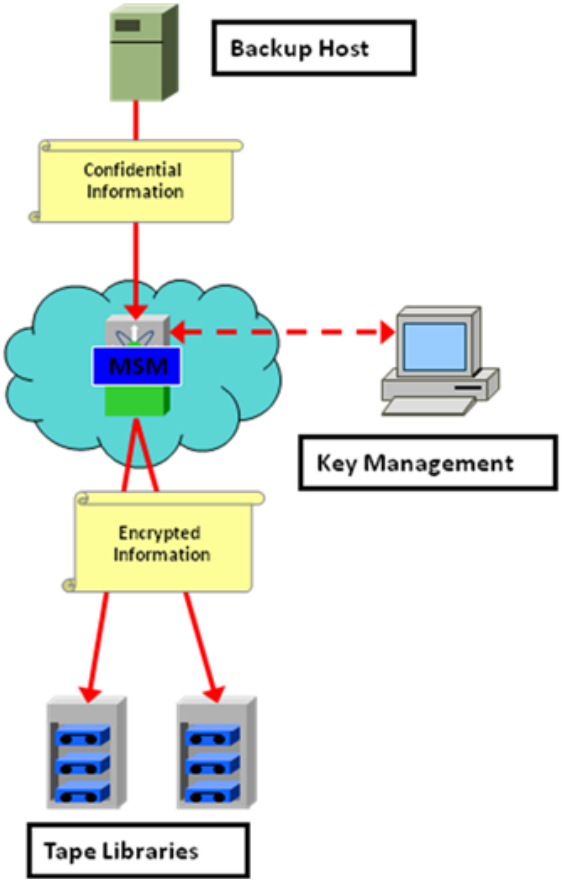


Figure 14 Cisco SME example

Supported key features

The following key features are supported for Cisco SME:

- Cisco switches support FC redirect feature that enables automatic redirection of traffic desired to be encrypted to an SME-enabled switch/module within the same fabric. The FC redirect feature gets away with the requirement of physical reconfiguration.
- Cisco MSM 18/4 modules (including the 9222i and SSN-16) use AES 256 encryption algorithm to protect data at rest. Cisco MDS 9000 NEX-OS supports advanced security features like Secure Shell (SSH), Secure Sockets Layer (SSL), RADIUS, and Fibre Channel Security Protocol (FC-SP) provide the foundation for the secure FIPS Level 3 architecture. Cisco SME uses the NIST approved random number standard to generate the keys for encryption.
- Cisco SME also features compression services which are enabled by default during encryption of data.
- Like other encryption appliances in the market, the SME offers configuration of role-based access control to the configuration and key management. The basic roles required for the SME operation are the Cisco SME Administrators and Cisco SME Recovery Officers. The capabilities and requirements of these roles change based upon the security level chosen during SME cluster configuration.
- The Cisco SME keys can be managed using the Cisco Key Management Centre (KMC) or the RSA key manager (RKM). The master key is protected using the smart cards. Quorum of (2 of 5) is enforced for recovery of master key.
- The Cisco SME supports the capability of clustering Cluster technology provides reliability and availability, automated load balancing, failover capabilities, and a single point of management.
- The SME cluster can be centrally configured and managed via the FM server which also interfaces with the key management server (CKMC/RKM).

Terminology

MSM (Multiservice Module) — Another name for the MDS-PBFI-1804 line card. Provides Fibre Channel, FCIP, iSCSI, FICON and SME functionality.

DPP (Data Plane Processor) — The encryption engine that resides on the MDS-PBFI-1804 (MSM).

ITL (Initiator-Target-LUN) also called an IT Nexus — The mapping of a host (initiator) tape device (target) and Logical Unit (LUN) in the SME environment.

Cluster — A collection of one or more switches that belong to the same SME environment.

Node — An individual MSM in a SME Cluster.

Keys — A code that is used to encrypt and decrypt a piece of information.

Topology guidelines

This section contains the requirements, general guidelines, sizing guidelines, and prerequisites for configuring SME.

Requirements

- Cisco SAN-OS 3.2.3x or later or NX-OS 4.1.3x or later as supported in the *EMC Support Matrix*
  - Must be installed on all switches that are running SME.
  - Switch connected to target device(s) should be a MDS-92xx/95xx series switch and running preferably similar firmware.
- Key Store
  - Encryption keys must be stored either in Cisco Key Manager Center (KMC) using either Postgres or Oracle database or in the RSA Key Manager Server (RKM) For supported versions, refer to the *EMC Support Matrix*.
  - Key stores of either solution should be configured in a High Availability (HA) configuration for redundancy, preventing unexpected data loss. In future releases higher than NX-OS v4.2.3, only the KMC will be supported.
- Fabric Manager Server
  - The FM Server must be installed and able to manage the fabric where SME is being configured.
    - The FMS version should follow the same version as the installed SME switch/fabric.
  - FM Standalone *cannot* be used.
  - FM License is not required for SME.
- Licenses
  - Each MSM or SSN-16 blade (not switch) requires an SME license for operation.

General guidelines

- All SME serviced tape libraries must be connected to a 9200 or 9500 series MDS switch.
- Switches must be running SAN-OS 3.2.3 or later or NX-OS 4.1.3x or later as supported in the *EMC Support Matrix*.
- At least one MSM or SSN-16 is required in each fabric that will utilize SME.Each SSN-16 has four encryption interfaces.
- An SME node can only belong to one cluster. Adding the node to more than one cluster will cause problems.
- MSM or SSN-16 modules should be as close to the targets as possible.
- Because of FC-Redirect limits, zone media servers and targets based on usage instead of all media servers to all targets.
- SME Clusters can span across multiple VSANs in a fabric.
- SME Cluster cannot span over multiple fabrics.

Sizing guidelines

- Each MSM supports up to 50 MB/s throughput with compression and encryption enabled (~4 Gb/s).
- For optimal performance each MSM or SSN-16 interface should be connected to 6-8 LTO-3 drives.
- Based on a single LTO-3 drive getting 40-60 MB/s throughput with compression and encryption enabled.
- Multiple MSMs can be configured in a single switch.
  - MDS-9506/9509/9513 can have multiple MSMs installed.
  - MDS-9222i can have two MSMs.
  - MDS-9216A/9216i can have one MSM.
- A fabric can have one SME Cluster, which can contain up to 4 switches.

Configuration limits

Table 6 lists the configuration limits.

Table 6 Configuration limits

Configuration	Limit
Number of switches in the fabric	10
Number of clusters per switch	1
Switches in a cluster	4
Fabrics in a cluster	2
Modules in a switch	11
Cisco MSM-18/4 modules in a cluster	32
Initiator-Target-LUNs (ITLs)	1024
LUNs behind a target	32
Host and target ports in a cluster	128
Number of hosts per target	128
Tape backup groups per cluster	2
Volume groups in a tape backup group	4
Cisco Key Management Center (# of keys)	32K
Targets per switch that can be FC-redirected	32

Prerequisites for configuring SME

- FM Server uses SSH to communicate with the DPP on the MSM or SSN-16.
  - SSHRSA keys must be configured on the switches housing the MSMs or SSN-16s.
  - SSH2 must be enabled on the switches or chassis housing the MSMs or SSN-16s.
- Cluster service must be enabled on all relevant switches.
- SME service must be enabled on all relevant switches.
- SME Interface must be created on each MSM.

- SME Interface is named SME <*line card#*>/<*port#*> (e.g., SME4/1, DNS must be configured on all SME switches as well as the FM Server).
- IF DNS is not used, the smeserver.properties file must be edited to reflect the DNS status.
- FM Server must be restarted after changing the file.
- Upgrading FM Server reverts the file to default.
- FM Server monitoring the fabrics that SME resides should be set to "Manage Continuously."

Core-Edge topology

In core-edge topology, media servers are at the edge of the network, and tape libraries are at the core. Figure 15 shows an example of a core-edge topology.

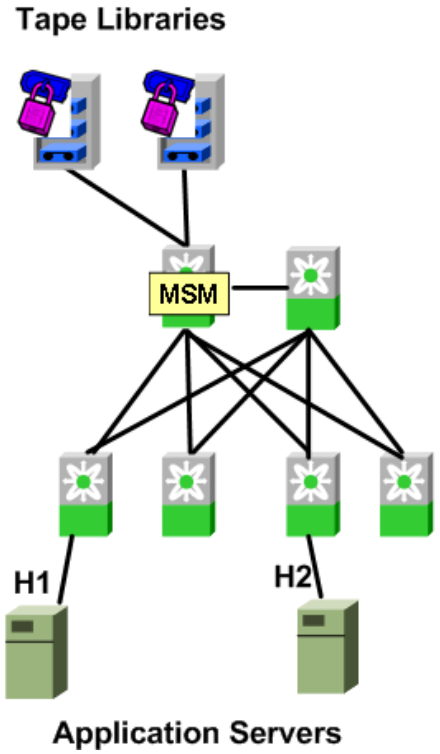


Figure 15 Core-edge topology example

If the targets that require SME services are connected to only *one* switch in the core, use MSM-18/4 or the SSN-16 modules and provision SME on this switch only. The number of MSM modules depends on the throughput requirements. If the targets that require SME services are connected to *multiple* core switches, connect MSM-18/4 or the SSN-16 modules and provision SME on these switches. Based on the throughput requirements, derive the total number of MSM-18/4 or the SSN-16 modules and spread them (in proportion to the expected traffic) across the switches where the targets are connected. Additionally, provision the ISLs between the target-connected switches in the core to account for SME traffic.

In Edge-Core-Edge topology, the hosts and the targets are at the two edges of the network connected via core switches. If the targets that require SME services are connected to only one switch on the edge, use MSM-18/4 or the SSN-16 modules and provision SME on this switch only. The number of MSM modules depends on the throughput requirements.

Single fabric SME cluster deployment

This section provides two examples.

- Supported single fabric deployment with RKM, NX-OS 4.2.3 and earlier (Figure 16 on page 100)
- Supported single fabric deployment with KMC used with SAN-OS and NX-OS (Figure 17 on page 101)

It also contains the following requirements:

- "Zoning requirements" on page 102
- "FC-Redirect requirements" on page 102
- "Configuration requirements" on page 102

Figure 16 on page 100 provides an example of the supported single fabric deployment of the Cisco SME along with the zoning, FC-Redirect, and configuration requirements used with firmware NX-OS v 4.2.3 and earlier.

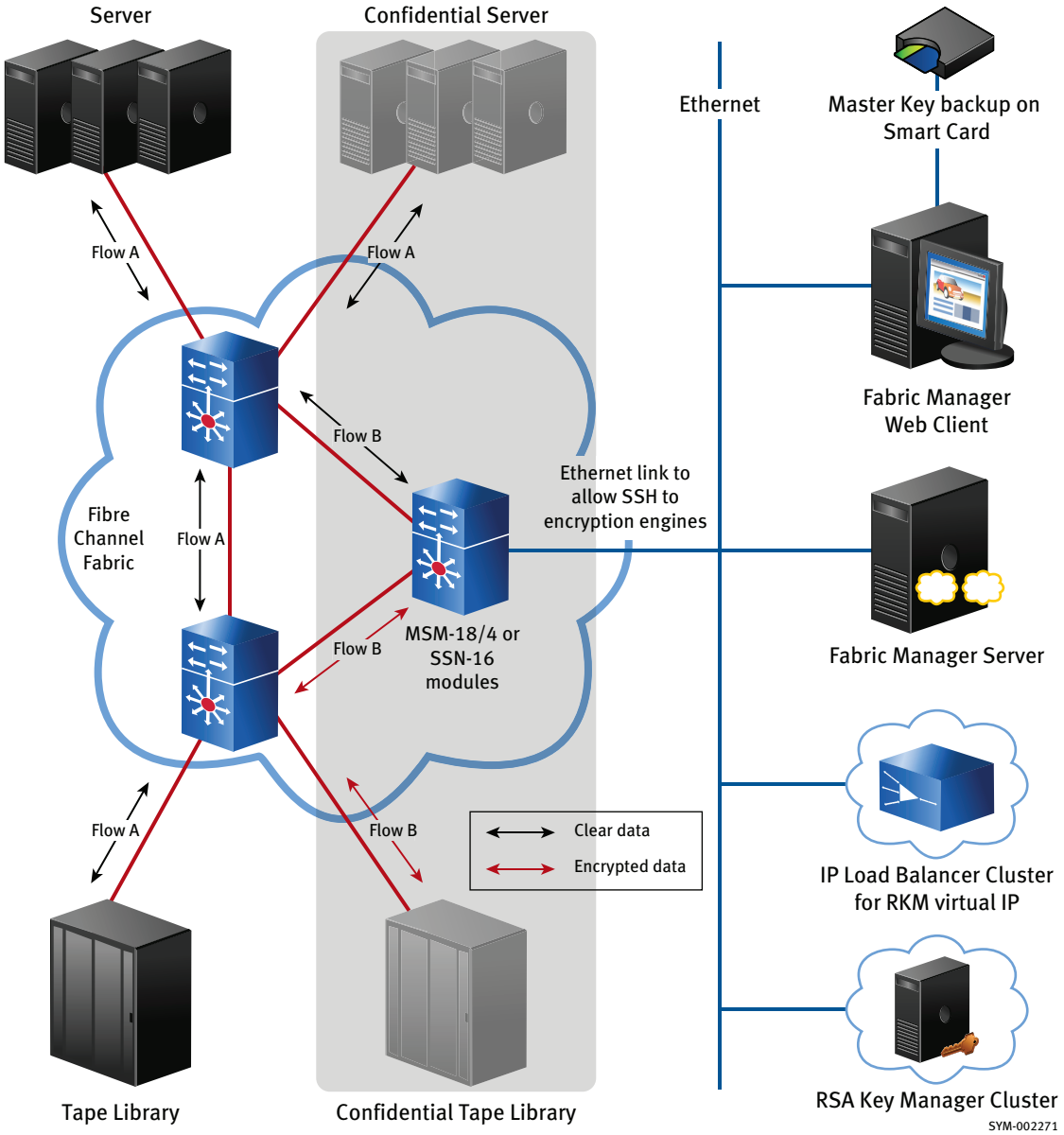


Figure 16 **Supported single fabric deployment with RKM, NX-OS 4.2.3 and earlier**

Figure 17 shows an example of supported single fabric deployment with KMC used with SAN-OS and NX-OS.

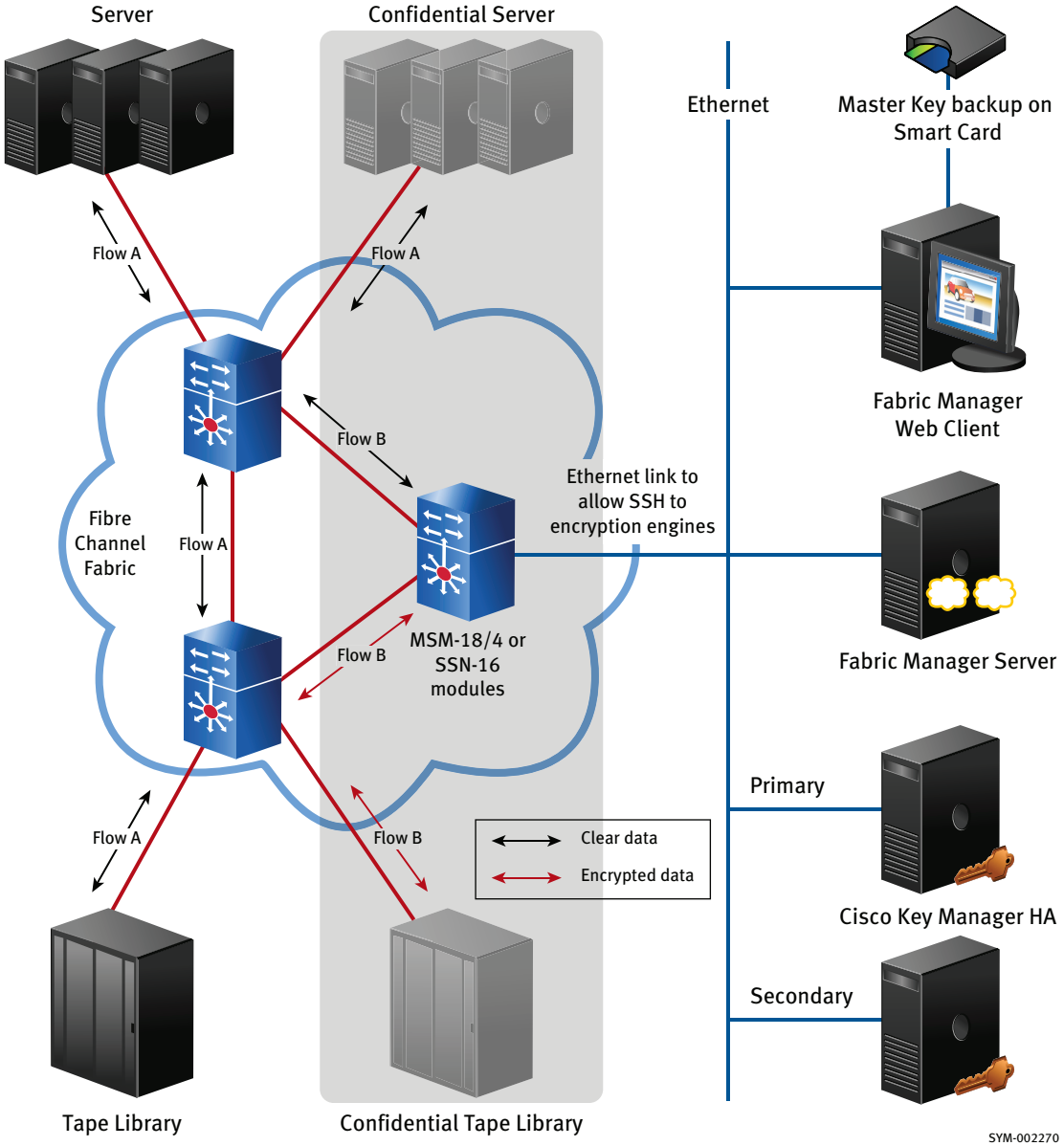


Figure 17 Supported single fabric deployment with KMC, SAN-OS and NX-OS

Zoning requirements

The following are zoning requirements:

- Default zone must be set to deny.
- Virtual N-Ports must not be zoned with any host or target.
- SME supports WWPN zoning only at this time.

FC-Redirect requirements

The following are FC-Redirect requirements:

- 32 Targets per MSM can be FC-Redirected.
- Each FC-Redirected target can be zoned with a maximum of 16 hosts.
- CFS should be enabled on all switches required for FC-Redirect.
- SME servers and tape devices should not be part of an IVR zoneset.
- SME must not be used in conjunction with SAN Device Virtualization (SDV) or Inter-VSAN Routing (IVR).

Configuration requirements

The following are configuration requirements:

- On an MSM, either iSCSI or SME can be configured, but not both simultaneously.
- Configurations are mutually exclusive and cannot coexist.
- Both use the same port indices.
- iSCSI and SME **can** be configured on different line cards in the same switch chassis.
- IVR cannot be enabled on SME enabled switches.
- FCIP Write Acceleration and FCIP Tape Acceleration must not be configured in the SME data flow.
- SME traffic between host and target must not pass through FCIP tunnels with FCIP WA and FCIP TA enabled.
- Avoid using FCIP and IPSec services on the same MSM that is running SME Services.

Key hierarchy overview

Cisco SME includes a comprehensive and secure system for protecting encrypted data using a hierarchy of security keys. Keys are essential to safeguarding your encrypted data and should not be compromised. Keys should be stored in the KMC or RKM. In addition, unique tape keys can be stored directly on the tape cartridge. The keys are identified across the system by a globally unique identifier (GUID).

This section contains the following information:

- "Key types" on page 103
- "Levels of security" on page 105
- "Key managers" on page 105
- "Key management best practice" on page 108
- "Cisco SME tape configuration" on page 108

Key types

The Cisco SME key management system includes the following types of keys:

- Master key

  The highest level key is the master key, which is generated when a cluster is created. Every cluster has a unique master key. Using key wrapping, the master key encrypts the tape volume group keys, which in turn encrypts the tape volume keys.

  When a Cisco SME cluster is created, a security engine generates the master key. Considering that a single fabric can host more than one cluster (for example, to support the needs of multiple business groups within the same organization) there will be as many master keys as there are clusters. Each master key is unique and is shared across all cluster members. The master key is used to wrap the tape volume group keys.

  For recovery purposes, the master key can be stored in a password-protected file or in one or more smart cards. When a cluster state is Archived (the key database has been archived) and you want to recover the keys, you will need the master key file or

the smart cards. The master key cannot be improperly extracted by either tampering with the MSM-18/4 module or by tampering with a smart card.

- Tape volume group keys

  The tape volume group key is used to encrypt and authenticate the tape volume keys, which are the keys that encrypt all tapes belonging to the same tape volume group. A tape volume group can be created on the basis of a bar code range for a set of backup tapes or it can be associated with a specific backup application. Tape volume group keys are occasionally rekeyed for increased security or when the security of the key has been compromised.

- Tape volume keys

  The tape volume key is used to encrypt and authenticate the data on the tapes.

  In unique key mode, the tape volume keys are unique for each physical tape and they can be stored in the Cisco KMC, RKM, or stored on the tape. The Cisco KMC or RKM database does not need to store a tape volume key if the key is stored on the tape itself. The option to store the key on the tape may dramatically reduce the number of keys stored on the Cisco KMC or RKM.

  In shared key mode, there is one tape volume key which is used to encrypt all volumes in a volume group.

Cisco SME interacts with the KMC or the RKM for managing the keys that are generated for encrypting data. The following components are used by the key management feature. The choice of key management solution (CKMC or RKM) cannot be changed once configured.

- SME Web Client — Provides the front-end GUI for SME configuration and key management. This is implemented as a part of the FM Web client and is launched by an RBAC role, such as the Security Administrator and the Recovery Officers, from their management stations.
- Smart Cards — Used only by the Recovery Officers. The smart card reader is attached to the management station on which the SME Web Client is launched.
- SME configuration and key management — Implemented as a part of the FM server. SME configuration module is responsible for cluster creation management and configuring the security policies for the backend storage and tape libraries. Key Management module is responsible for managing the backup and

archive of the data encryption key catalog. The key catalog database may be implemented in a database in the FM Server itself or can be integrated into an external oracle database in the organization.

Levels of security

Cisco SME cluster supports the following three levels of security for backing up the cluster's master key upon creation:

- Basic — The master key is stored in a password protected key file.
- Standard — The master key is stored on a single smart card.
- Advanced — The master key shares are written to five smart cards. Two or three smart cards are required to restore

The only location whereby keys are stored in plain text will be within the crypto engine for encryption and decryption operations. This will be further discussed in Chapter 5, "Cisco SME Configuration in a Cisco Key Manager Environment."

Both the tape volume group key and the tape volume key must be backed up and stored on a key manager to facilitate the proper key management lifecycle. This ensures a proper centralized or clustered store of keys that can be managed either through the key manager (RKM) or through the SME web client interface.

Key managers

SME allows a choice of the key manager RKM or KMC when the Fabric Manager Server is configured on its first run. This choice is permanent and only a complete purge of the database can allow this selection to be changed.

Cisco Key Management Center

The Cisco Key Management Center (Cisco KMC) is the centralized management system that stores the key database for active and archived keys. The keys stored in the Cisco KMC are not usable without the master key. To manage the potential increase in tape volume keys, Cisco SME provides the option to store the tape volume key on the tape itself. In this case, the Cisco KMC stores the tape volume group keys.

This option exponentially increases the number of managed tapes by reducing the number of keys stored on the Cisco KMC. However, this option also restricts the capability of purging keys at a later time.

The Cisco KMC provides the following advantages:

- Centralized key management to archive, purge, recover, and distribute tape keys
- Integrated into Fabric Manager Server depending on the deployment requirements.
- Integrated access controls using AAA mechanisms.

Cisco Key Manager (KMC) supports two different setups:

- KMC is integrated with Cisco Fabric Manager, as shown in Figure 18. Key exchange traffic and management traffic will go directly to the FMS with integrated KMC.

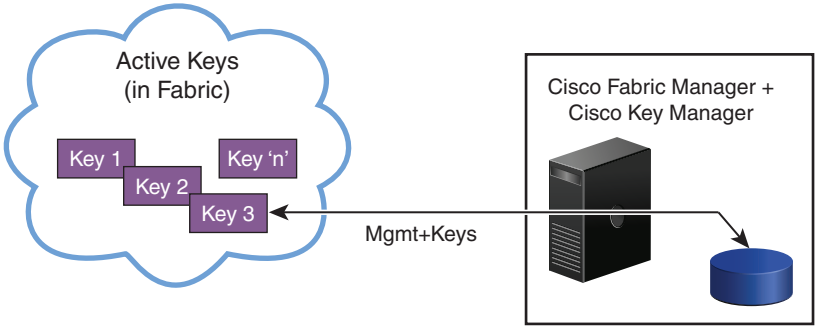


Figure 18 KMC integrated with Cisco Fabric Manager

- KMC is decoupled from Cisco Fabric Manager, as shown in Figure 19. Key exchange traffic goes directly to the KMC while fabric management traffic goes directly to the FMS.

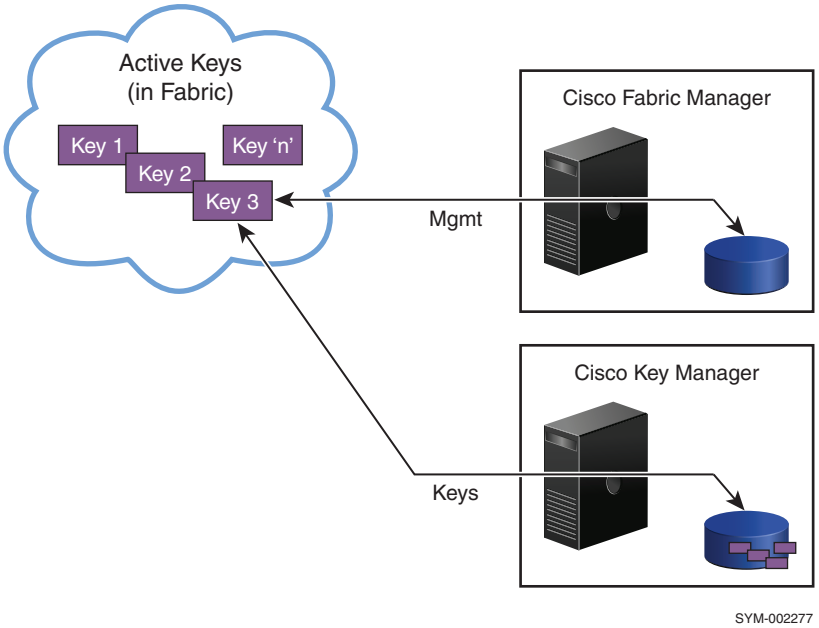


Figure 19 KMC decoupled from Cisco Fabric Manager

RSA key manager

The RSA key manager is an easy-to-use, centrally administered encryption key management system that can manage encryption keys at the database, SAN, tape, and disk storage layer. It is designed to simplify the deployment and ongoing use of encryption throughout the enterprise.

Always refer to the *EMC Support Matrix* for the proper RKM and NX-OS/SAN-OS version interoperability.

Suggested topologies for RKM HA can be found in Chapter 2, "Implementing RSA Key Manager (RKM) HA Functionality."

Figure 20 shows how tape keys can be stored in the RKM.

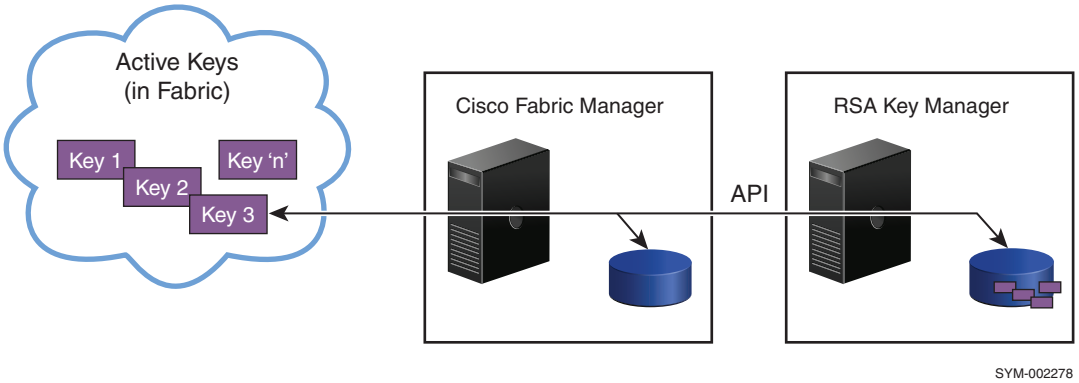


Figure 20 Storing the keys using RKM example

Key management best practice

It is always considered a best practice to store the tape keys away from the data they protect. However, since tape is a removable medium and due to the limitation of the number of keys supported by a key management server, the customer might opt to store the tape keys on the tape itself. This can be configured when the SME cluster is created. The choice must be made based upon the security policies and resource restrictions of the organization.

Cisco SME tape configuration

The Cisco SME provides tape configuration to enable the user to define the paths tha t need to be encrypted. This helps users separate the attached media into tape groups or volume groups and to also filter out the media based upon the regex functionality. The following are the basic classifications defined under tape configuration.

- Tap e g roup — A backup environment in the SAN. This consists of all the tape backup servers and the tape libraries that they access.
- Tape device — A tape drive that is configured for encryption.
- Tape volume — A physical tape cartridge identified by a barcode for a given use.

- Tape volume group — A logical set of tape volumes configured for a specific purpose. Using Cisco SME, a tape volume group can be configured using a barcode range or a specified regular expression. In an auto-volume group, a tape volume group can be the volume pool name configured at the backup application.

Cisco SME provides the capability to export a volume group with an encryption password. This file could later be imported to a volume group. Also, volume group filtering options provide mechanisms to specify what type of information will be included in a specific volume group. For example, you could filter information in a volume group by specifying a barcode set or date range.

The encryption tape group for Celerra NDMP protocol is to be configured using the CLI. Follow the steps given in the *Cisco SME Configuration Guide*, located at www.cisco.com.

Implementation best practices

Consider the following best practices:

- In the first implementation there may be some small overhead associated with encryption. It will vary with the data set type but a 10% worst case should be considered.
- If the encryption switch is placed behind an EMC CDL, physical tapes being restored should be done using the direct import mode. This method uses the application software to do the restore.
- Not all tape drives and applications are supported. Make sure you reference Cisco's support matrix.
- Volumes are either *clear text only* or *encrypted only.*

5

Cisco SME Configuration in a Cisco Key Manager Environment

This chapter discusses Cisco SME configuration for enhanced security and HA in a Cisco Key Manager environment. The following topics are discussed:

- Overview 112
- SAN 113
- Key management 114
- IP network 119

Overview

The Cisco SME solution can be readily set up and deployed due to its design, which allows SAN-based encryption to tape by simply adding a supported encryption node into the existing Cisco fabric. This chapter explores further security settings that can be configured to better suit the customer environment based on how restrictive the security policies are in place.

SAN

SAN covers the Fibre Channel connectivity from the host to the target. This section is concerned with the encryption to tape portion in the SME solution. It is also in the SAN where the encryption node resides and Fibre Channel frames are redirected to it upon the creation of tape groups. Tape groups enforce encryption to tape of a host initiator to a tape target through the use of Fibre Channel frame redirection of traffic from the host to the encryption node before travelling to its target.

Proper zoning best practices of having a single initiator and target in a zone and ensuring that the zoneset is activated prior to creating a tape group should be followed. This facilitates the proper access control of hosts that will have access to the data on the tapes.

The tape encryption cryptographic algorithm used in SME is AES-CBC-256 and is not user configurable.

Key management

The key management encompasses three areas that can be configured, each discussed further in this section:

- "Key Manager" on page 114
- "Master key security selection" on page 116
- "Tape media specific key settings" on page 117
- "Tape recycling" on page 118

Key Manager

The Cisco Key Manager Center (KMC) offers many deployment strategies depending on how highly available it is expected to be deployed. The KMC is installed as part of the Fabric Manager Server (FMS) installation, which means that a copy of KMC would be installed with every current FMS installation. This means that you can have an all-in-one box that consists of the FMS plus KMC that simultaneously manage the fabric and provide key management facilities, or be highly available and decouple all the servers into separate redundant boxes. This section will discuss the most highly available solution.

When deploying a KMC, you need to understand that it should be used in conjunction with the FMS and when planning its deployment you must also consider the FMS as part of the SME total solution. These consist of the following core components:

- FMS — The Fabric Manager Server that discovers and continuously manages the fabric that SME resides on.
- FMS database — The FMS database consists of everything else in a typical FMS database besides encryption keys.
- KMC — The decoupled Key Manager Center, which is actually a FMS installation but dedicated for key management purposes.
- KMC database — The KMC database, which main purpose is to store encryption keys.

Typically, for a non-federated FMS deployment, the FMS database is installed together with FMS using the default PostgreSQL database. In the solution, it is not so crucial for these two components to be HA because only performance data might be lost if this server fails.

The KMC HA solution consists of a pair of KMC servers that provides high availability and reliability. These high availability servers help to avoid both downtime and loss of data through synchronization and redundancy. The key management solution consists of a primary and a secondary KMC server, which point to the same database.

Both the KMC servers should use the same Oracle 11g Enterprise installation to achieve high availability. The Oracle 11g Enterprise installation should be installed on the two servers and synchronized using Oracle Active Data guard.

Each SME cluster is configured with primary and secondary KMC servers. The primary server is preferred over the secondary server. The cluster is connected to the primary server and, at any indication of failure, connects to the secondary server. The cluster periodically checks for the availability of the primary server and resumes connection to the primary server when it becomes available. All the switches in a cluster use the same KMC server. When a switch connects to a secondary server, an automatic cluster-wide failover occurs to the secondary KMC server. The switches in the cluster fail back to the primary KMC server once it is available.

There is minimal configuration required on both the primary and secondary KMC servers. They only need to select that they are functioning as a Cisco Key Manager Center and enter in the IP address of the associated primary or secondary role of the peer.

Another important consideration when deploying the Key Manager is to ensure that all KMC hosts involved, including the backend networks to the database cluster, are secured. This is beyond the scope of this documentation and configurations are highly specific to the organization. The following list are some resources that provide guidelines for hardening servers to meet FIPS, security considerations in a Microsoft Windows environment, and Oracle database best practices:

- NIST — Computer Security Resource Center
  http://csrc.nist.gov/index.html
- Microsoft security — http://www.microsoft.com/Security/
- Oracle — Security Information and Best Practices
  http://www.oracle.com/security/

Master key security selection

Whenever there is a cluster creation, a new master key will be generated for it. The purpose of the master key is to wrap (encrypt) the volume group keys or encrypted volume keys before storing them in the key manager. It is important that the master key is backed up before any encryption operation takes place. The cluster will be operational only until the master key is backed up.

The following options are available upon cluster creation:

- Basic — The master key is stored in a file and encrypted with a password. To retrieve the master key, you need access to the file and the password.
- Standard — Standard security requires one smart card. When you create a cluster and the master key is generated, you are asked for the smart card. The master key is then written to the smart card. To retrieve the master key, you need the smart card and the smart card pin.
- Advance — Advanced security requires five smart cards. When you create a cluster and select Advanced security mode, you designate the number of smart cards (two or three of five smart cards or two of three smart cards) that are required to recover the master key when data needs to be retrieved. For example, if you specify two of five smart cards, then you will need two of the five smart cards to recover the master key. Each smart card is owned by a Cisco SME Recovery Officer.

The basic option uses a file to store the keys while the standard and advance options use a smart card. All the options store the master key encrypted through either a password protected file when the basic option is chosen or PIN protected smart card(s) access when the standard or advance option is chosen. Only the advance option allows quoru-based recovery, which might make sense for organizations requiring multiple stakeholders holding a portion of a key for the data being protected.

During normal operation of the cluster, the master key resides in plain text within the cryptographic module of the encryption node. This master key is persistent, even through reboots. The only time that a master key can be restored is when a cluster status is deactivated.

Tape media specific key settings

The keys that actually perform encryption to tapes are known as volume tape keys. Upon creation of a cluster, the security administrator will be presented with a few options pertaining to how these keys are created, stored, and recycled, as shown in Table 7.

Table 7 Key settings (page 1 of 2)

	Description	Security	Considerations
Shared	In shared key mode, only tape volume group keys are generated. All tape volumes that are part of a tape volume group share the same key.	Medium to low. A compromise to one tape volume group key will compromise the data in all tapes that are part of that same tape volume group.	**Cisco KMC key database**—Is smaller storing only the tape volume group keys. **Purging**—Available only at the volume group level Considerations for choosing this can include: • Limited storage space for KMC database together with the fact that these physical tapes are frequently handled by 3rd party hence not wanting encrypted keys to be store in tape. • The tape volumes contain similar data sets hence no difference in security if one tape is compromised.
Unique Key	In unique key mode, each individual tape has it's own unique key. The default value is enabled.	High. A compromise to a tape volume key will not compromise the integrity of data on other tape volumes.	**Cisco KMC key database**—Is larger storing the tape volume group keys and every unique tape volume key. **Purging**—Available at the volume group and volume level. The default setting and recommended for most cases. However do bear in mind the amount of tape media that will be deployed so as to provision appropriately the KMC database.

Table 7 Key settings (page 2 of 2)

	Description	Security	Considerations
Unique Key with Key-On-Tape	In the key-on-tape mode, each unique tape volume key is stored on the individual tape. You can select key-on-tape (when you select unique key mode) to configure the most secure and scalable key management system.. The default value is disabled. **Note:** When key-on-tape mode is enabled, the keys stored on the tape media are encrypted by the tape volume group wrap key.	Medium to high. A compromise to a tape volume key will not compromise the integrity of data on other tape volumes.	**Cisco KMC key database—** Increases scalability to support a large number of tape volumes by reducing the size of the Cisco KMC key database. Only the tape volume group keys are stored on the Cisco KMC. **Purging—**Available at the volume group level. Considerations for this would be it eases the management of tape volume group keys as tape volume keys are never stored on the KMC. However, if the physical tapes regularly exchange hands, there is a possibility that the data on this tape volume might be compromised.

Tape recycling

There is a further setting to determine the retention of keys when tape volume groups are destroyed. If tape recycling is enabled, old keys for the tape volume are purged from Cisco KMC when the tape is relabeled and a new key is created and synchronized to the Cisco KMC. This setting should be selected when you do not need the old keys for previously backed-up data that will be rewritten.

The default setting is **Yes**. Setting this option to No is required only if tape cloning is done outside of the Cisco SME tape group.

IP network

This section discusses how to enhance the underlying IP network's security used for switch management and how the encryption nodes communicate encryption keys with the KMC. It includes the following topics:

- "Securing the management of switches to the FMS" on page 119
- "Securing the web client communications to the FMS" on page 120
- "Securing the MDS to KMC communication through SSL" on page 121

Securing the management of switches to the FMS

On the Cisco MDS switches, the default SNMP user that FMS uses to communicate with the switch uses a weak authentication and no privacy is required. The SNMP hosts that the MDS sends traps to also uses SNMP v2c with a community string. These are rather weak default protocols that should be changed before connection with a FMS.

To alter the SNMP user credentials for better security:

```
Switch (config)# snmp-server user admin auth sha password priv aes-128 password
```

This example command creates the snmp user admin with the authentication algorithm chosen as "sha", the desired password (shown here as "password," but you should use a secure one), and the privacy algorithm as AES-128 with the desired password (also shown here as "password").

To alter the SNMP host trap for better security:

```
Switch (config)# snmp-server host 10.10.10.1 traps version 3 priv admin udp-port
   2162
```

This example command creates an SNMP v3 trap to host 10.10.10.1 with authPriv security level choosing SNMP user "admin" through the notification host's UDP port 2162.

Doing so allows a more secure management communications between MDS and the FMS. This, however, does not affect how the FMS will communicate with the MDS when performing SME type operations, such as viewing the SME clusters or creating clusters.

Such communicates are always performed through a SSHv2 tunnel. Therefore, enabling SSH is a prerequisite before SME can be used.

Securing the web client communications to the FMS

The preferred interface used to configure SME is through the Fabric Manager web client This is because it is the only interface to configure master key security using smart cards. It is therefore a prerequisite to ensure that the SSL option is checked when installing FMS or KMC.

FMS will only allow its own self-signed certificates to be installed when the SSL option is checked. There is no other option during installation to edit or use an alternative certificate and trust pair.

If the organization where SME is being deployed employs an in-house CA or uses a trusted third-party CA signing service, then it is preferred that this SSL connection also be configured to be trusted.

The certificates required are similar to what SSL secured web servers use: the server's own signed credential certificate and the trustpoint certificate. In the context of SME, it is required that these certificates reside within a Java keystore or truststore, such as the following:

- fmserver.jks — A Java keystore containing the certificate bearing the public and private key signed by a trusted CA.
- fmtrust.jks — A Java truststore containing the certificate bearing the public key of the trusted CA.

Creating a Java keystore certificate

To create a certificate signing request, sign it with a CA, then create a Java keystore certificate, by completing the following steps:

1. Create an RSA keypair.
2. Create a Certificate Signing Request (CSR) by associating it with the previously created keypair (pay attention to the Common Name "CN," which by convention is the host name).
3. Send the CSR to be signed by a CA to retrieve the signed certificate.
4. The signed certificate and its private key should be exported into a PKCS12 format certificate containing public and private keys.
5. The Java keystore can then be created by importing the PKCS12 certificate.

These certificates are then used to replace the current self-signed certificates.

For the updated and detailed configuration steps, refer to the *Cisco Storage Media Encryption Configuration Guide* at www.cisco.com.

Securing the MDS to KMC communication through SSL

In order for the MDS to write or read encrypted tape volumes, it requires either the volume group key or volume key, depending on its operation. This requires the MDS to make a TCP connection to the KMC for such operations.

By default, such connections are not secured at the TCP connection. Remember that the encryption keys are wrapped by its higher key hierarchy, whereby the master key resides at the highest level. These keys are transmitted as XML messages and key data are secured at the application layer. To further improve the security of this connection, we can optionally enable SSL.

When SSL is used to secure a link, it provides two forms of security:

- Authentication
- Privacy

For authentication and privacy to be valid, a trusted CA is required, which will sign certificate credentials of trusted hosts or switches. However, it is quite common that organizations do not implement an in-house CA or use a trusted third-party signer but still requires the privacy provided by SSL encryption. This is where the organization can self-sign the certificates and still provide connection confidentiality. In summary:

- In-house CA or trusted third-party — Enables both Authentication and Privacy
- Self-signing — Privacy

To enable SSL, it is required that all KMC hosts and encryption capable switch nodes be provisioned with a its signed credential certificate and the same trustpoint certificate.

The steps of creating a signed certificate are discussed in "Creating a Java keystore certificate" on page 120.

For updated and detailed configuration steps, refer to the *Cisco Storage Media Encryption Configuration Guide* at www.cisco.com.

6 Cisco Key Management Center (KMC) Migration Procedure

This chapter discusses a two site disaster recovery issue and provides a solution to ensure key availability and data integrity. Topics include:

- Issue and solution overview 124
- Step 1: Migration from two unique KMCs 127
- Step 2: Periodic backup and restore of the database 137
- Step 3: Disaster recovery procedure 138

Issue and solution overview

This section discusses a two-site disaster recovery issue and provides a solution to ensure key availability and data integrity.

Issue

Your current configuration has two unique instances of Key Management Centers (KMCs) on two sites. You require the database of the two KMCs to be in sync in order to serve as a backup site in case of a disaster scenario. Currently, it is not possible to merge the two KMC databases.

Solution

To ensure key availability and data integrity, complete the following steps, each discussed in more detail as noted:

- **Step 1:** Migrate from 2 KMCs to a warm standby KMC solution.

  Detailed steps are provided in "Step 1: Migration from two unique KMCs" on page 127.

- **Step 2**: Perform periodic backup of active KMC and restores on all standby KMCs.

  Detailed steps are provided in "Step 2: Periodic backup and restore of the database" on page 137.

- **Step 3:** Activate the standby KMC in case of failures with the active KMC and/or active cluster.

  Detailed steps are provided in "Step 3: Disaster recovery procedure" on page 138. Two case scenarios are provided in this step: "Case 1: KMC-A failure" on page 138 and ""Case 2: Complete site failure, KMC-A and SME-A cluster " on page 143.

For simplicity, the following terminology is used to refer to the entities in Site A and Site B.

Site A Site A contains the following:

RKM server1:	RKMA-1
RKM server2:	RKMA-2
Backup server:	Server-A
KMC:	KMC-A
SME cluster:	Cluster A
Tape library:	TL-A
Fabric Manager:	FM-A
F5 Cluster:	F5-A

Site B Site B contains the following:

RKM server3:	RKMB-1
RKM server4:	RKMB-2
Backup server:	Server-B
KMC:	KMC-B
SME cluster:	Cluster B
Tape library:	TL-B
Fabric Manager:	FM-B
F5 Cluster:	F5-B

Assumption "Step 1: Migration from two unique KMCs" on page 127 assumes that site B KMC (KMC-B) will be in the *standby* mode at the end of the migration procedure.

Prerequisites

The following prerequisites are *mandatory* for the solution to function as desired:

- Every site should have the same authentication details, including username, password, Radius settings, postgresql database passwords, switch authentication details, snmp-server, etc.
- All sme clusters must be created *only* by the active KMC,
- You must reinstall Fabric Manager on both sites with version 3.3.3.

  Note: Using any later versions would require reinstalling Fabric Manager version 3.3.3 which will cause loss of all key and database information on both sites.

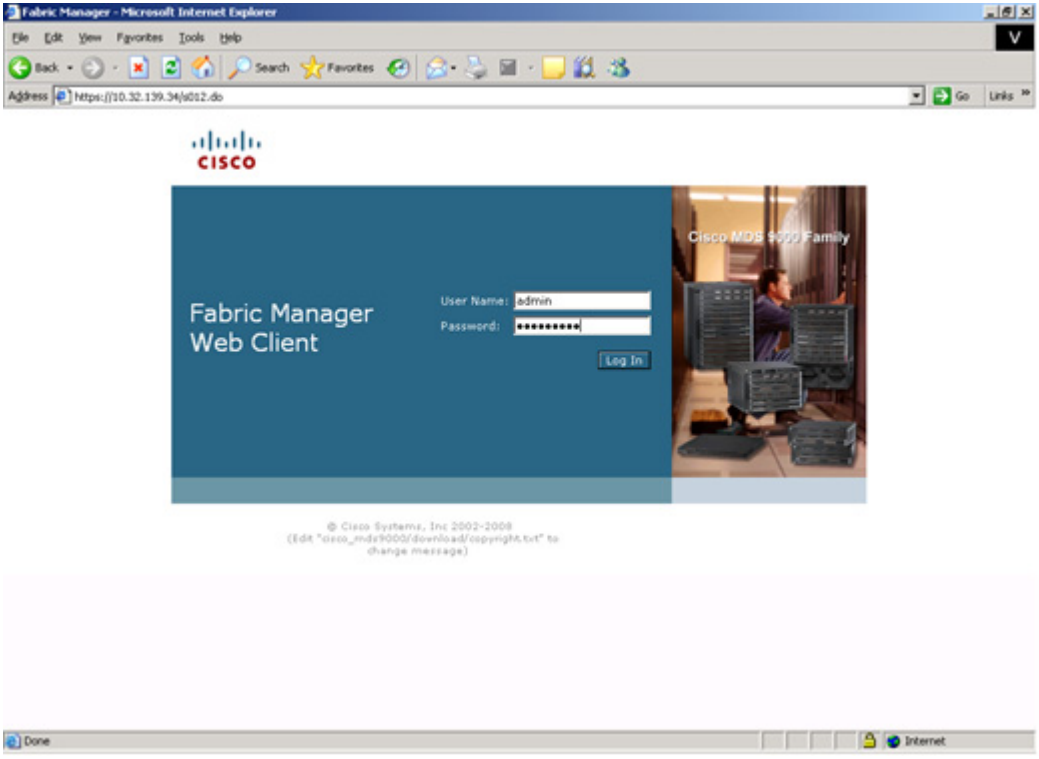
- All fabrics must be accessible to the active KMC.

Step 1: Migration from two unique KMCs

To migrate from two unique KMCs to a warm standby KMC solution, complete the following steps:

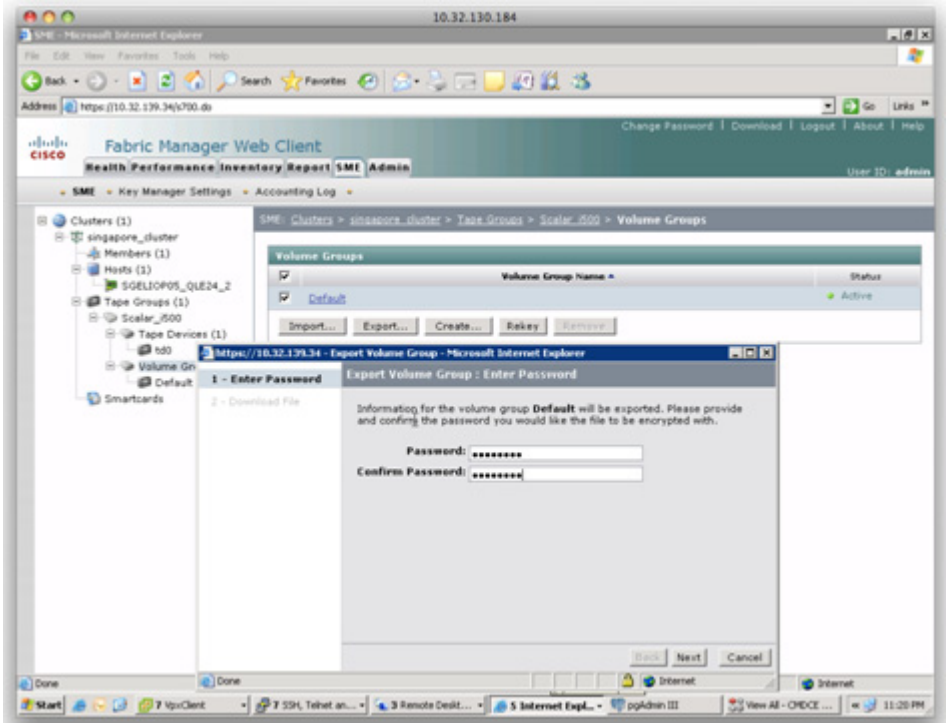
1. Log in to FM-B.

Note: KMC-B will be the standby KMC in the final configuration.



2. Confirm that the FM-B version is 3.3.3.
3. Export any tape groups that need to be imported into the new configuration.

Note: It is recommended to always perform a **pgbackup** command on the standby KMC *before* restoring the configuration on the standby KMC. To restore the configuration, use the **pgrestore** command.

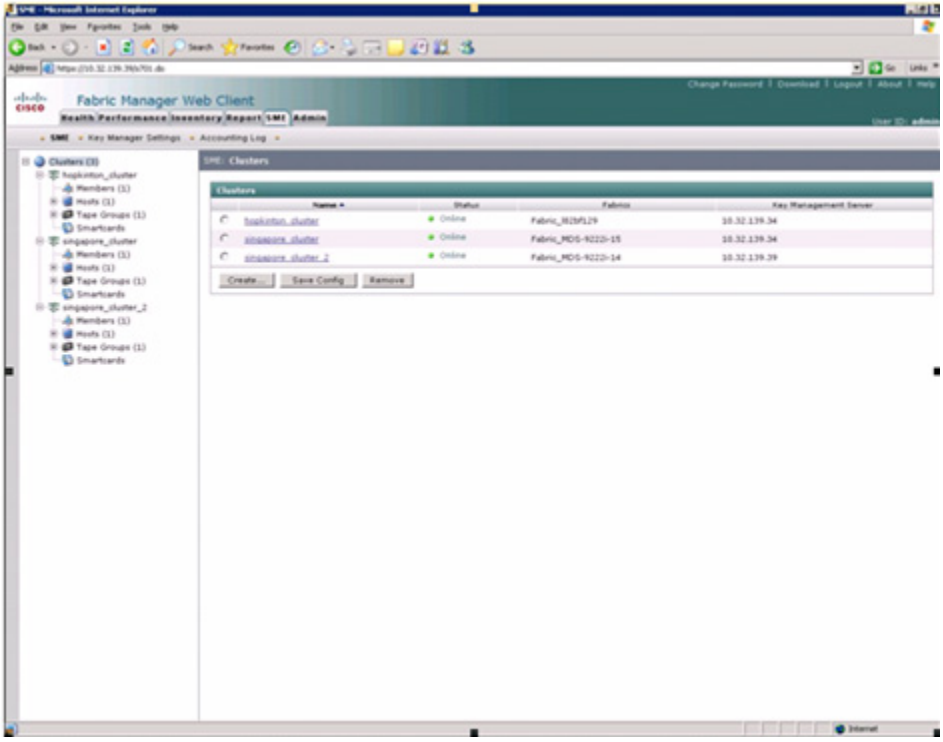


4. Delete the existing cluster/clusters managed by KMC-B.
5. Log in to FM-A.
6. Confirm the FM-A is running version 3.3.3.

IMPORTANT

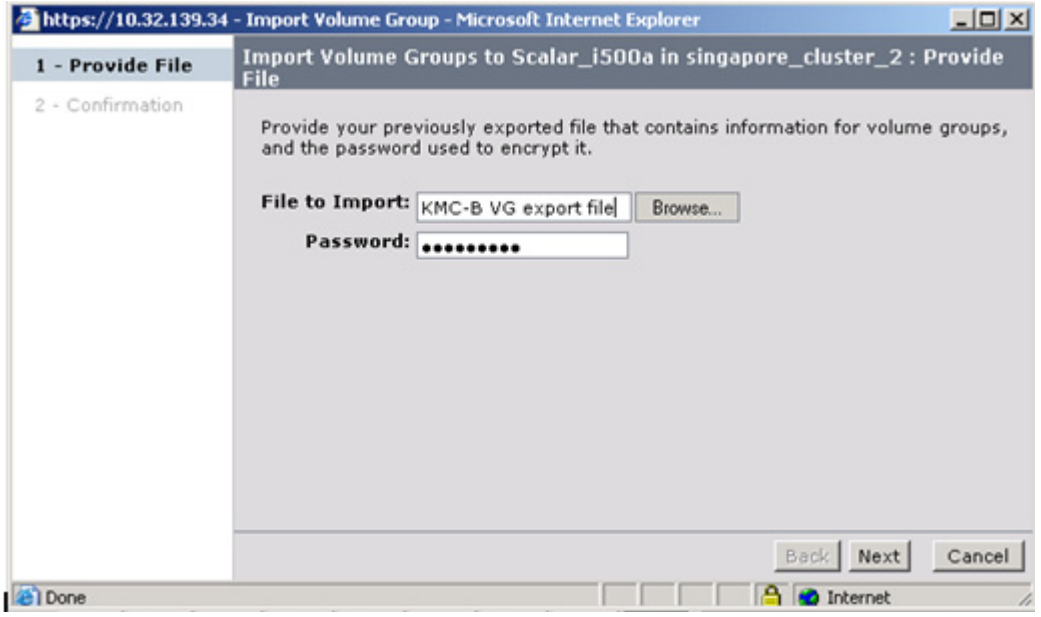
KMC-A will be used to manage all sme clusters across all sites. Create all sme clusters using the KMC-A admin GUI.

7. Verify that all clusters are online on KMC-A.



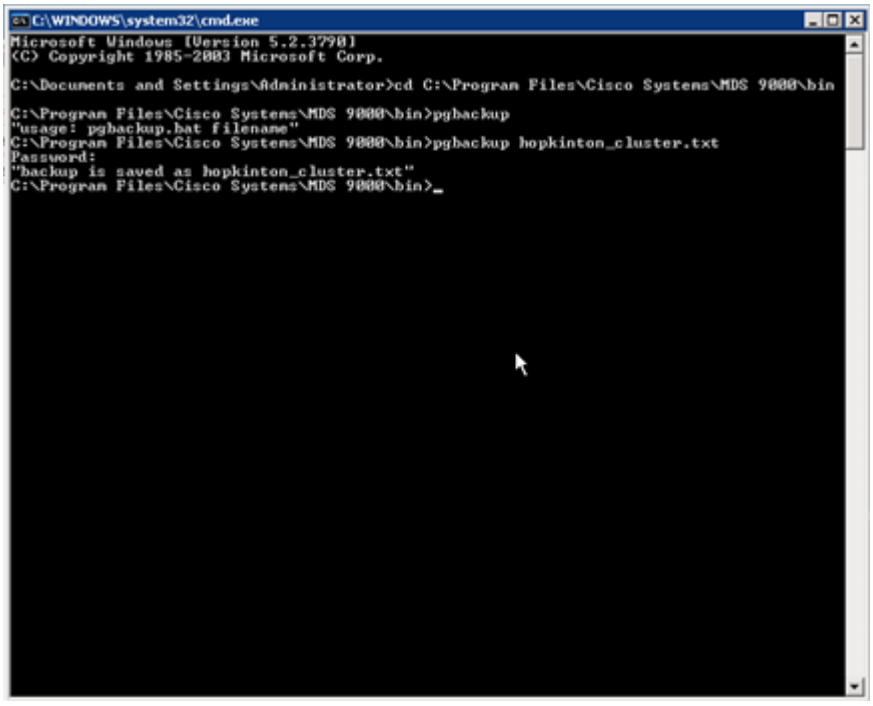
8. Import any volume group(s) into KMC-A that was previously exported from KMC-B (version 3.3.3) into the appropriate cluster(s).

Note: Any keys imported from a KMC that have been decommissioned will be in the archived state (read-only mode).

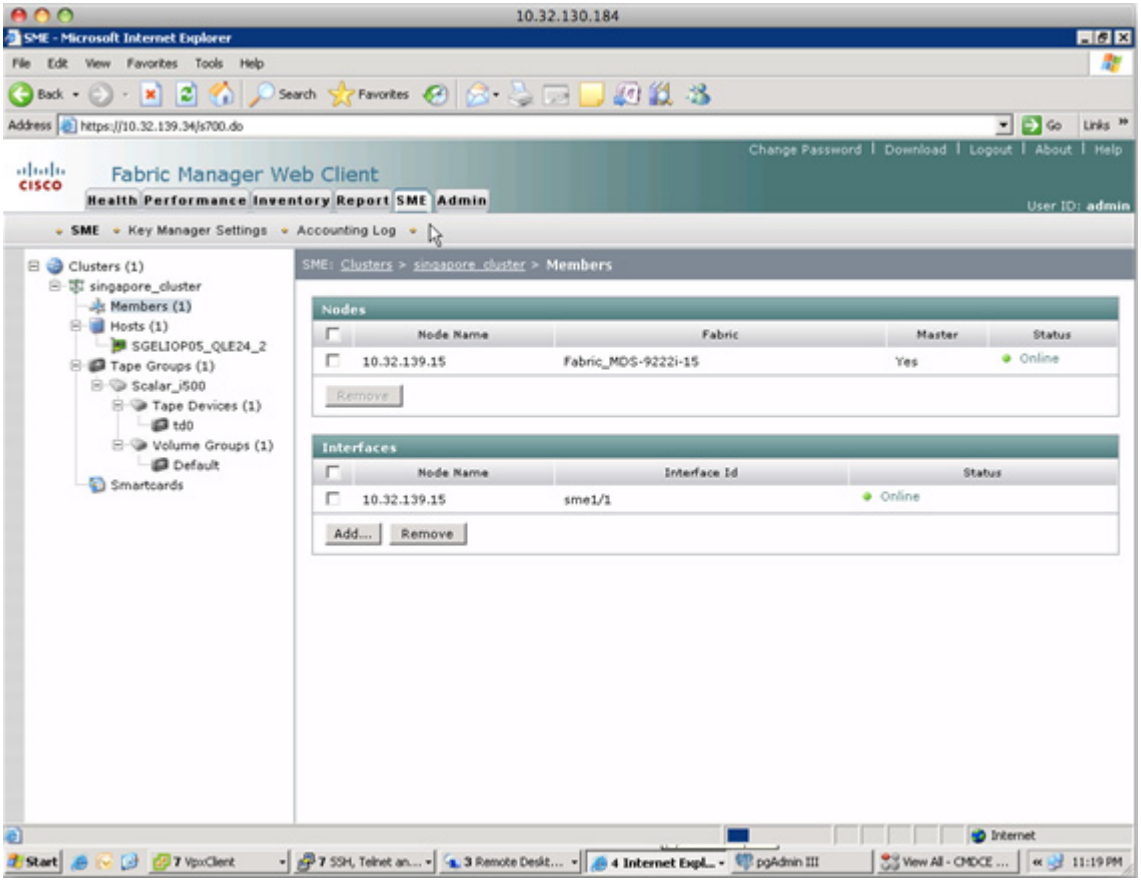

https://10.32.139.34 - Import Volume Group - Microsoft Internet Explorer
1 - Provide File
2 - Confirmation
Import Volume Groups to Scalar_i500a in singapore_cluster_2 : Provide File
Provide your previously exported file that contains information for volume groups, and the password used to encrypt it.
File to Import:
KMC-B VG export file
Browse...
Password:
Back
Next
Cancel
Done
Internet

9. Back up the KMC-A database using the **pgbackup** script under C:\Program Files\Cisco Systems\MDS 9000\bin:

   pgbackup <*backup file* >

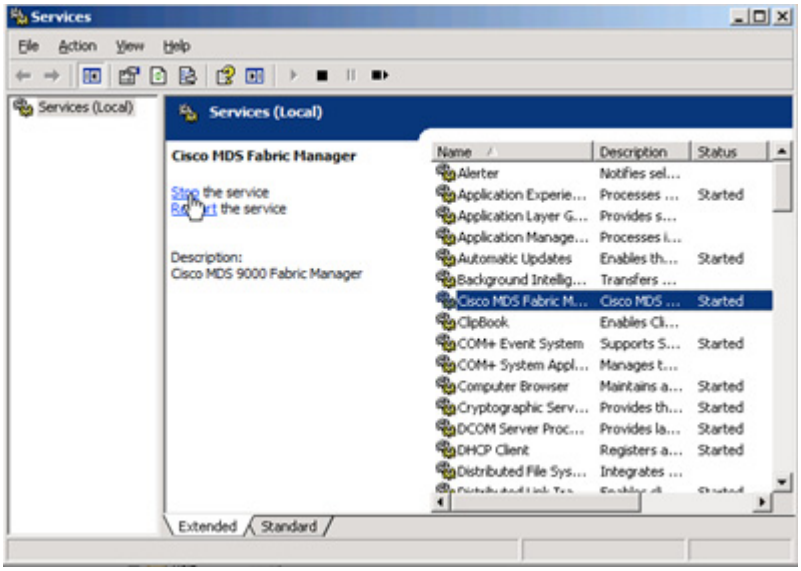


10. On KMC-A, perform **pgbackup**, using the following syntax:
pgbackup [*backup file*]



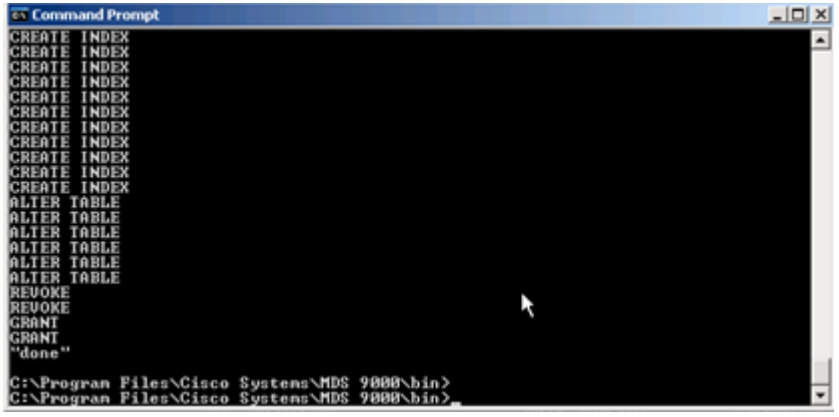
11. Copy the backup file from KMC-A to KMC-B.

12. Log in to the FM-B server and stop the Cisco MDS Fabric Manager Service.

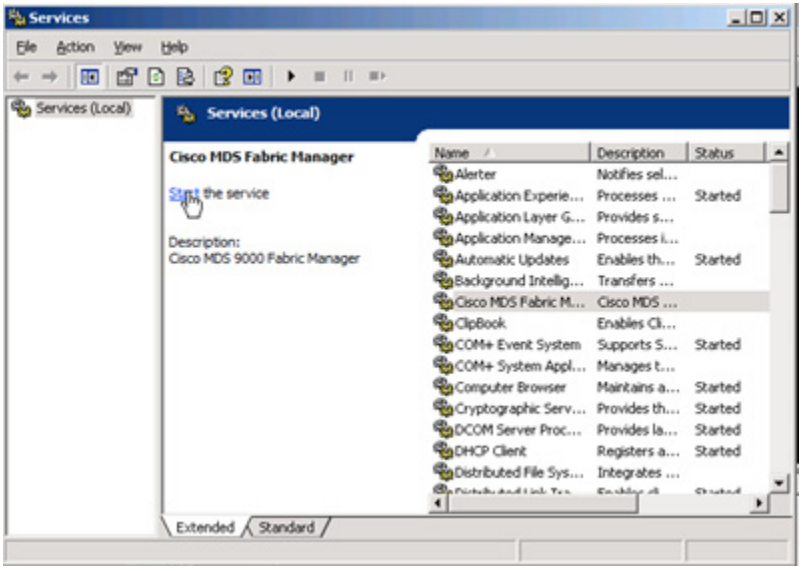


13. Restore the backed up file on KMC-B using the **pgrestore** script under C:\Program Files\Cisco Systems\MDS 9000\bin:

    opgrestore <*backup file*>



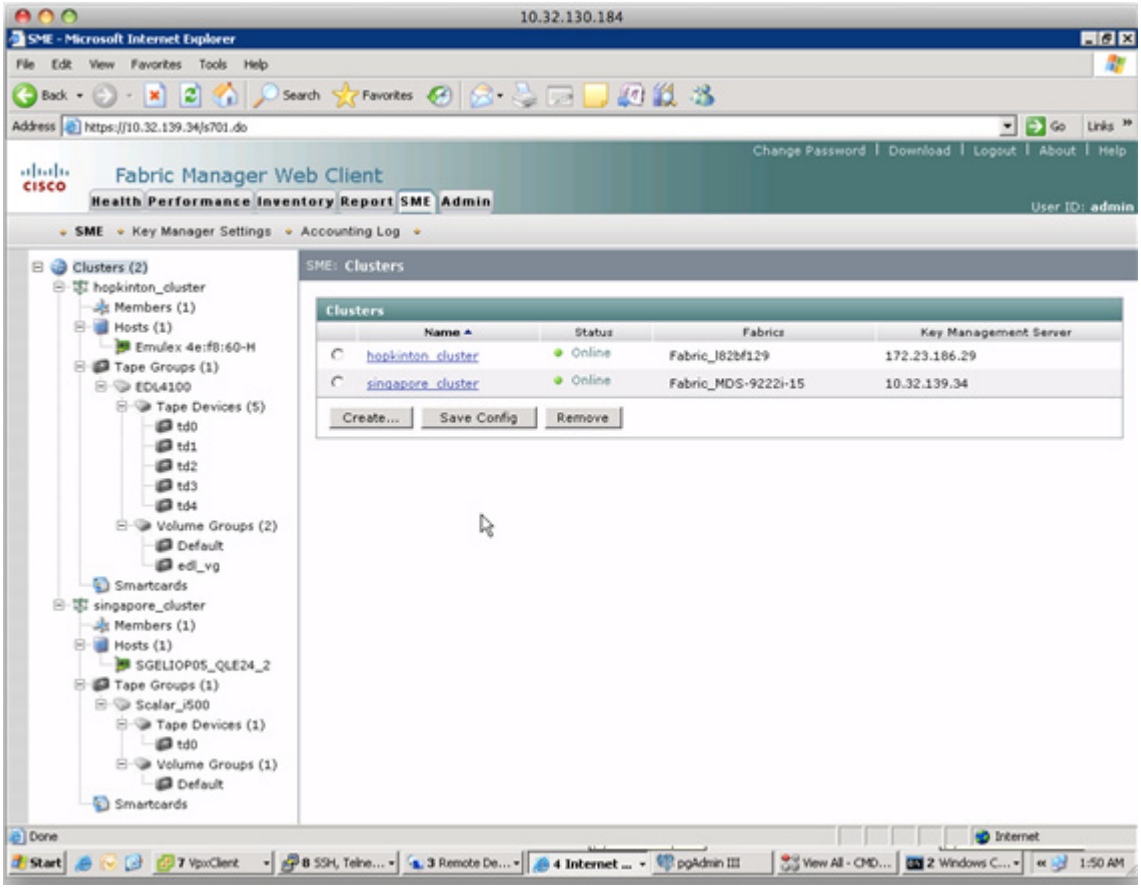
14. Start the Cisco MDS Fabric Manager service.



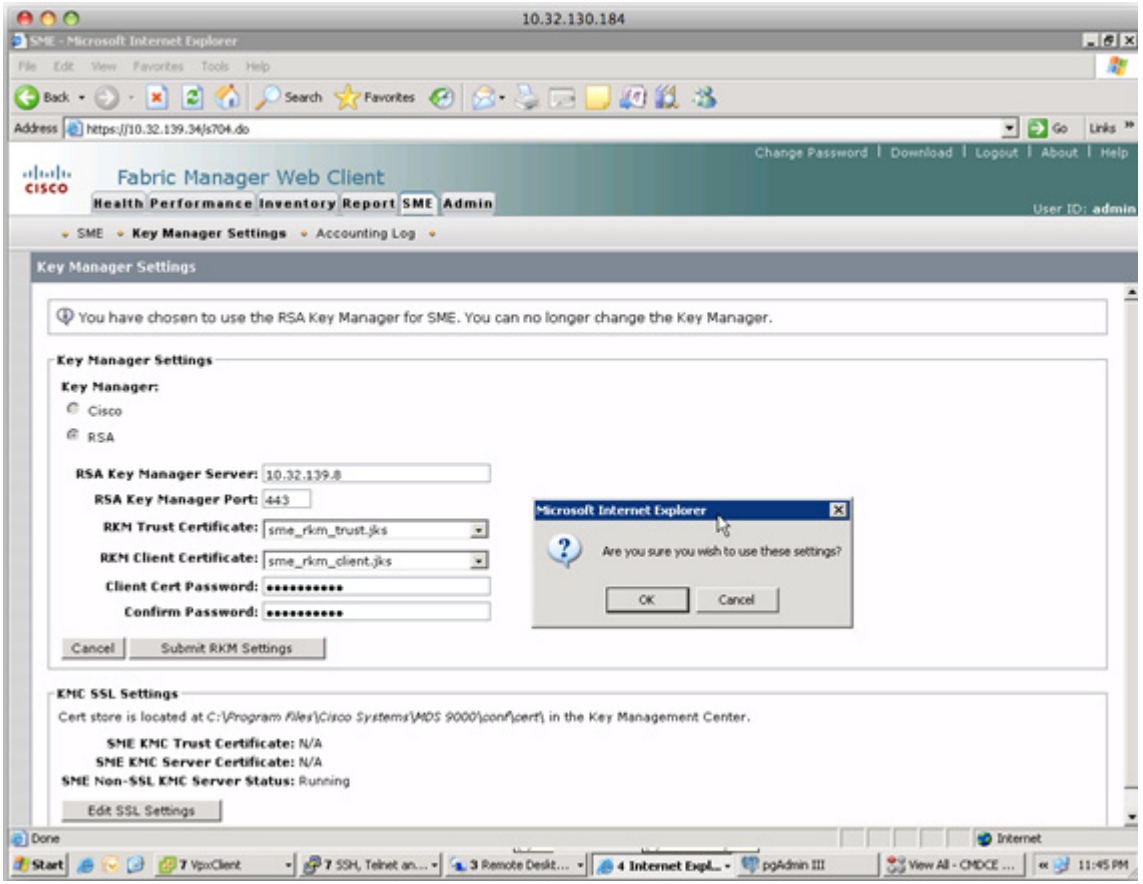
15. Verify that all fabrics are visible from KMC-B.

16. Verify all sme clusters are online on the KMC-B GUI.

Note: KMC-B is a warm standby server. All sme clusters will only communicate with KMC-A for any key transactions. This will ensure that all key transactions are directed through the active KMC (KMC-A).



The KMC-B configuration will contain the IP address of the F5-A under key manager settings. You can change this to reflect the ip address of F5-B for consistency. This setting comes into play *only* when KMC-B takes over as the active KMC.


10.32.130.184
SME - Microsoft Internet Explorer
Address https://10.32.139.34/s704.do
Change Password | Download | Logout | About | Help
Fabric Manager Web Client
Health Performance Inventory Report SME Admin
User ID: admin
SME Key Manager Settings Accounting Log
Key Manager Settings
You have chosen to use the RSA Key Manager for SME. You can no longer change the Key Manager.
Key Manager Settings
Key Manager:
Cisco
RSA
RSA Key Manager Server: 10.32.139.8
RSA Key Manager Port: 443
RKM Trust Certificate: sme_rkm_trust.jks
RKM Client Certificate: sme_rkm_client.jks
Client Cert Password:
Confirm Password:
Cancel
Submit RKM Settings
Microsoft Internet Explorer
Are you sure you wish to use these settings?
OK
Cancel
KMC SSL Settings
Cert store is located at C:\Program Files\Cisco Systems\MDS 9000\conf\cert\ in the Key Management Center.
SME KMC Trust Certificate: N/A
SME KMC Server Certificate: N/A
SME Non-SSL KMC Server Status: Running
Edit SSL Settings
Done
Internet
Start
11:45 PM

Step 2: Periodic backup and restore of the database

The **pgbackup** procedure (Step 9, page 131), and the **restore** procedure (Step 13, page 133) have been documented as a part of "Step 1: Migration from two unique KMCs" on page 127.

It is recommended that you run the **pgbackup** as a cronjob to ensure that you have the latest copy of the KMC database.

Pg restore is an offline process. Fabric Manger service must be stopped while restoring the database on the standby server.

Step 3: Disaster recovery procedure

Step 3 consists of the following two case scenarios and provides information on how to obtain disaster recovery in these situations:

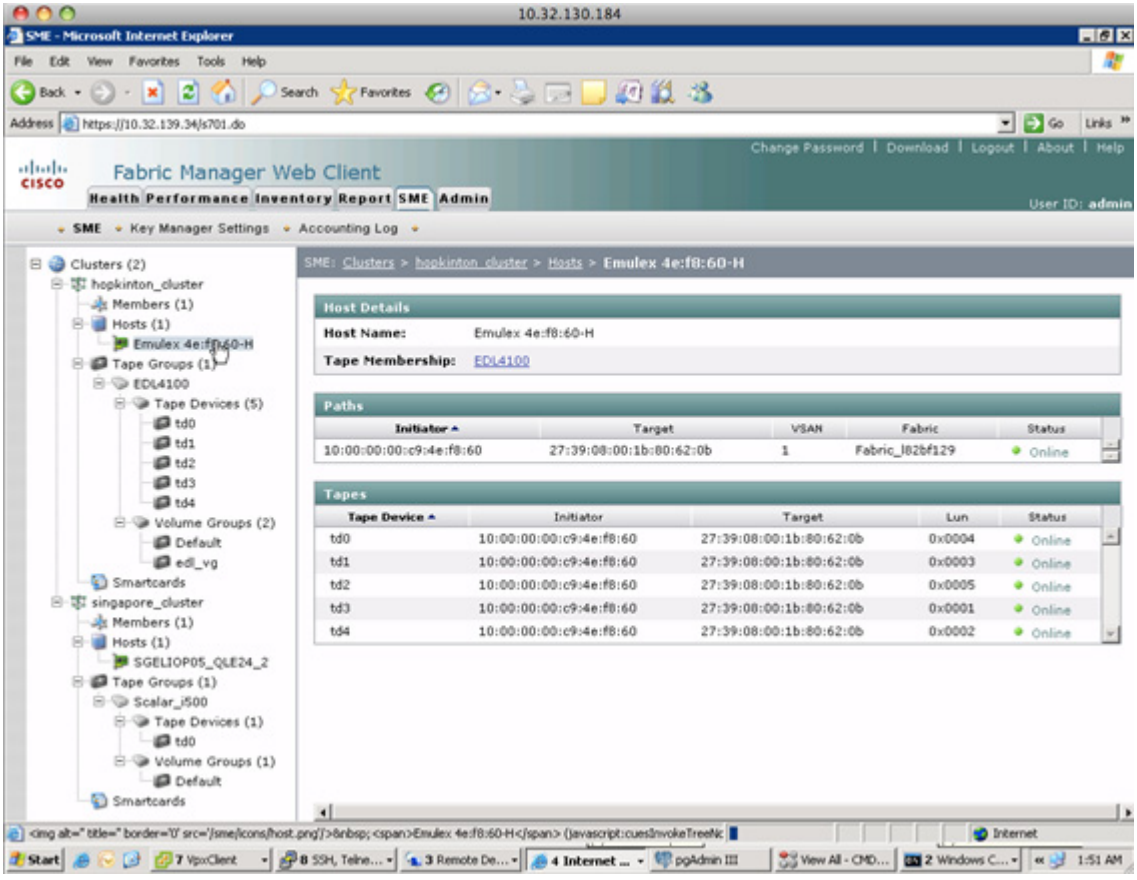
- "Case 1: KMC-A failure" on page 138
- "Case 2: Complete site failure, KMC-A and SME-A cluster " on page 143

Case 1: KMC-A failure

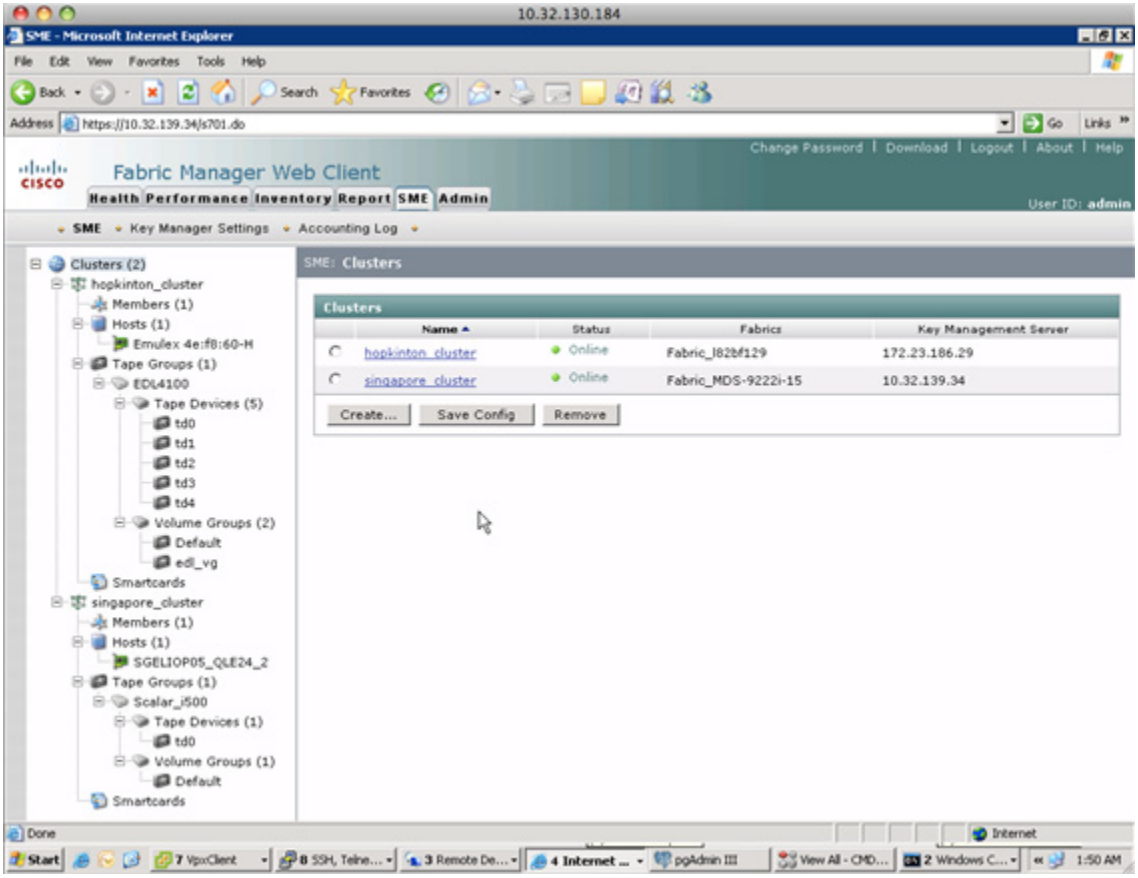
If the active KMC-A fails on Site A, you must activate KMC-B located at Site B as the new active KMC. To accomplish this, complete the following steps:

1. Log in to KMC-B.
2. Perform the restore procedure as documented in Step 13, page 133, to ensure that the latest copy of backup from KMC-A is applied to Site B.
3. Verify that all fabrics are visible from KMC-B.

4. Verify all targets and hosts are online.



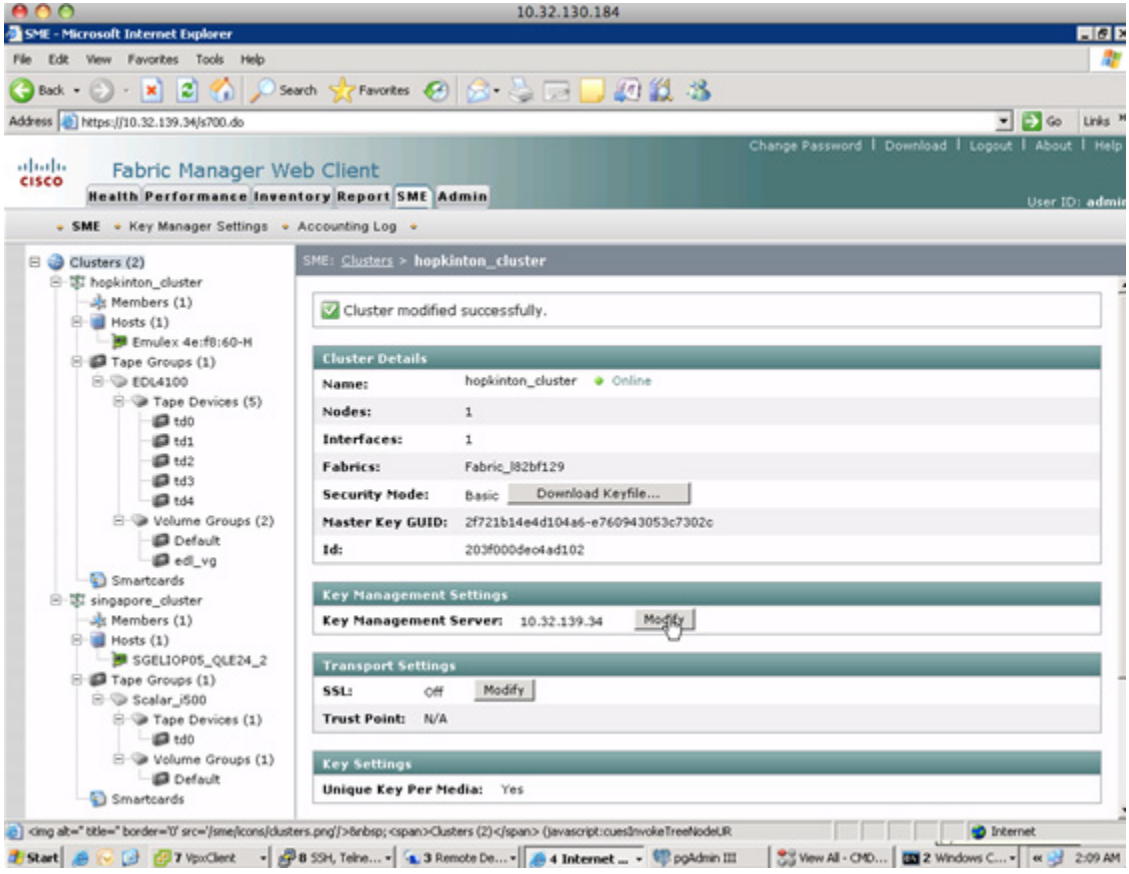
5. Verify all sme clusters are online on the KMC-B GUI.



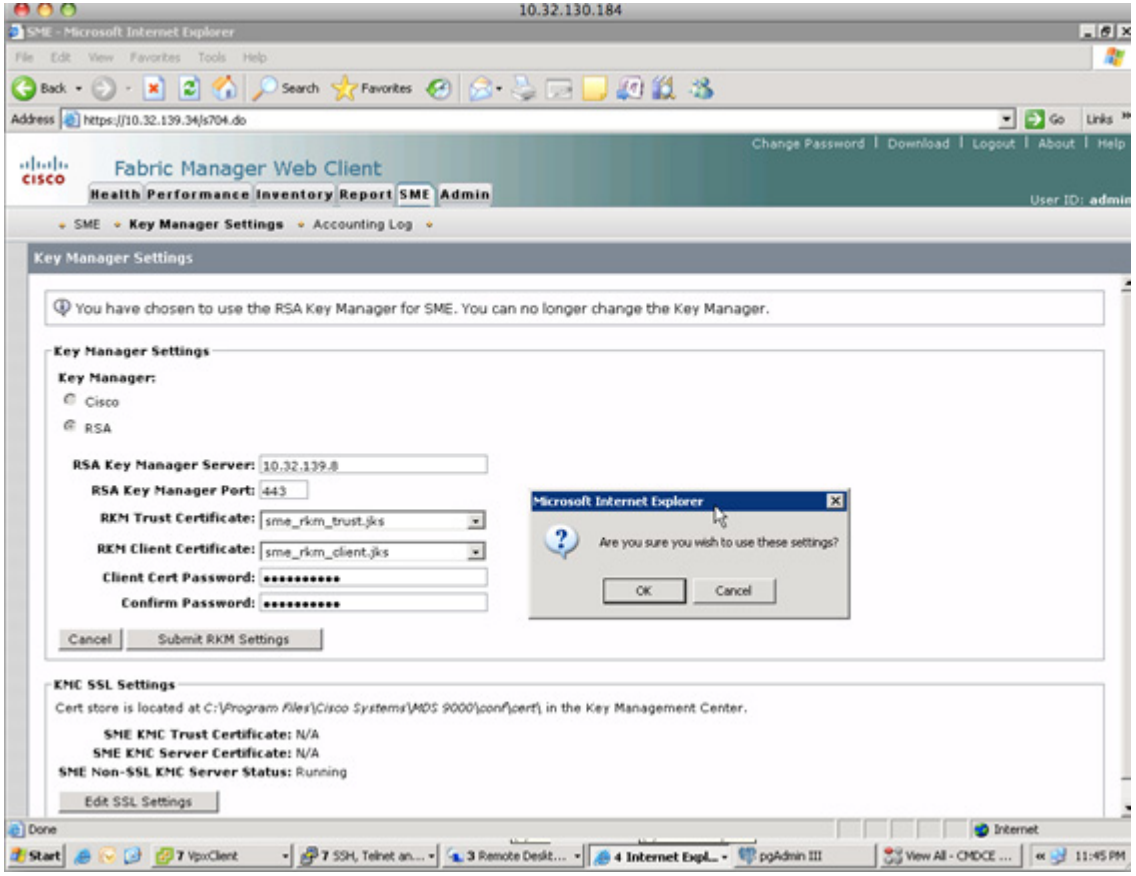
6. Change the KMC IP address on all sme clusters to point to KMC-B.

Note: All sme clusters now need to communicate using KMC-B for any key transactions. This will ensure that all key transactions are directed through the active KMC (KMC-B).

Do *not* restart KMC-A until all sme clusters point to KMC-B. This will prevent clusters from storing new information on a standby KMC (KMC-A).



7. Change the RKM server setting in KMC-B to point to F5-B.


10.32.130.184
Fabric Manager Web Client
Health Performance Inventory Report SME Admin
SME Key Manager Settings Accounting Log
Key Manager Settings
You have chosen to use the RSA Key Manager for SME. You can no longer change the Key Manager.
Key Manager Settings
Key Manager:
Cisco
RSA
RSA Key Manager Server: 10.32.139.8
RSA Key Manager Port: 443
RKM Trust Certificate: sme_rkm_trust.jks
RKM Client Certificate: sme_rkm_client.jks
Client Cert Password:
Confirm Password:
Cancel
Submit RKM Settings
Microsoft Internet Explorer
Are you sure you wish to use these settings?
OK
Cancel
KMC SSL Settings
Cert store is located at C:\Program Files\Cisco Systems\MDS 9000\conf\cert\ in the Key Management Center.
SME KMC Trust Certificate: N/A
SME KMC Server Certificate: N/A
SME Non-SSL KMC Server Status: Running
Edit SSL Settings

KMC –B is now the active KMC.

The database on KMC-B *must* be backed up on a periodic basis to capture any changes made to the configuration.

In the event that you plan to migrate back to KMC-A, the latest backup of the database from KMC-B must be restored back to KMC-A as a part of the migration procedure detailed in "Step 1: Migration from two unique KMCs" on page 127.

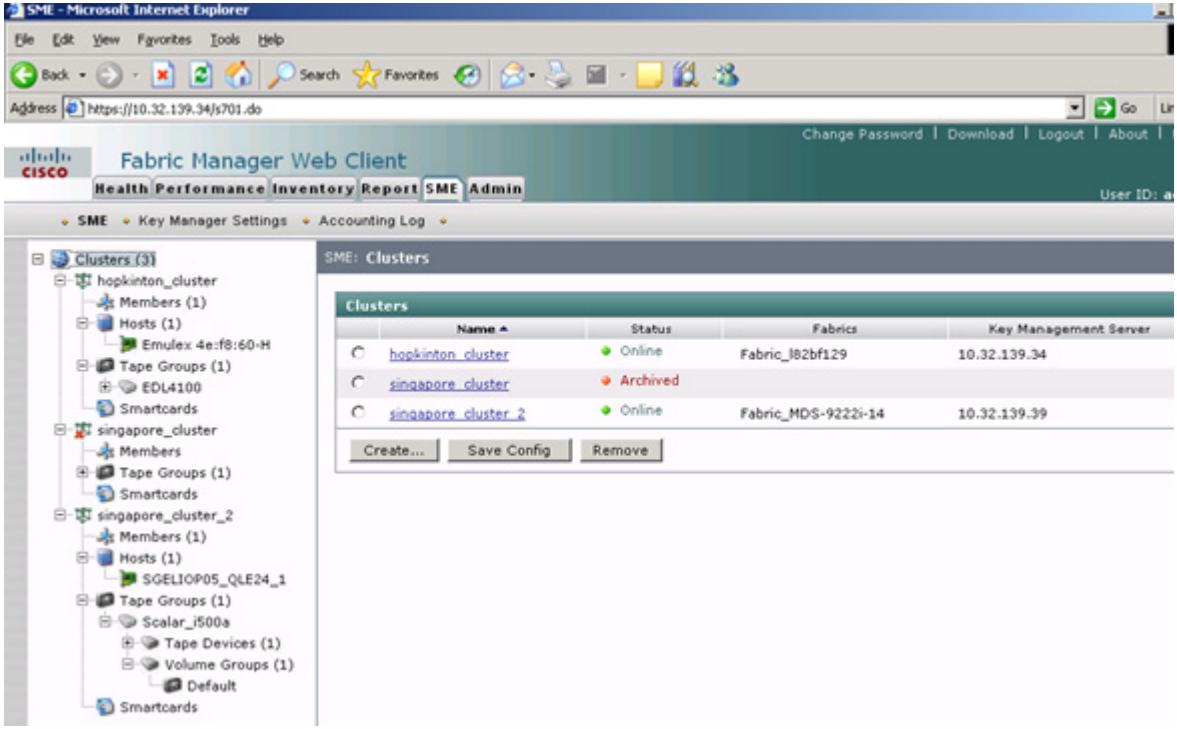
Case 2: Complete site failure, KMC-A and SME-A cluster

In case where there is complete site failure, the following procedure will activate KMC-B as the active KMC and enable access to the encrypted data backed up on tapes using SME-A. To accomplish this, complete the following steps:

1. Log in to KMC-B.
2. Perform the restore procedure explained in Step 13, page 133 to ensure that the latest copy of backup from KMC-A is applied to site B.
3. Check the status of all fabrics and clusters.

   All clusters expect the one affected by the disaster (SME-A) should be online. The affected cluster will be in the *archived* state.

Note: The KMC can take up to 10 minutes to update the status of the failed sme cluster. Do *not* proceed further until the status of the affected cluster changes to *archived*.



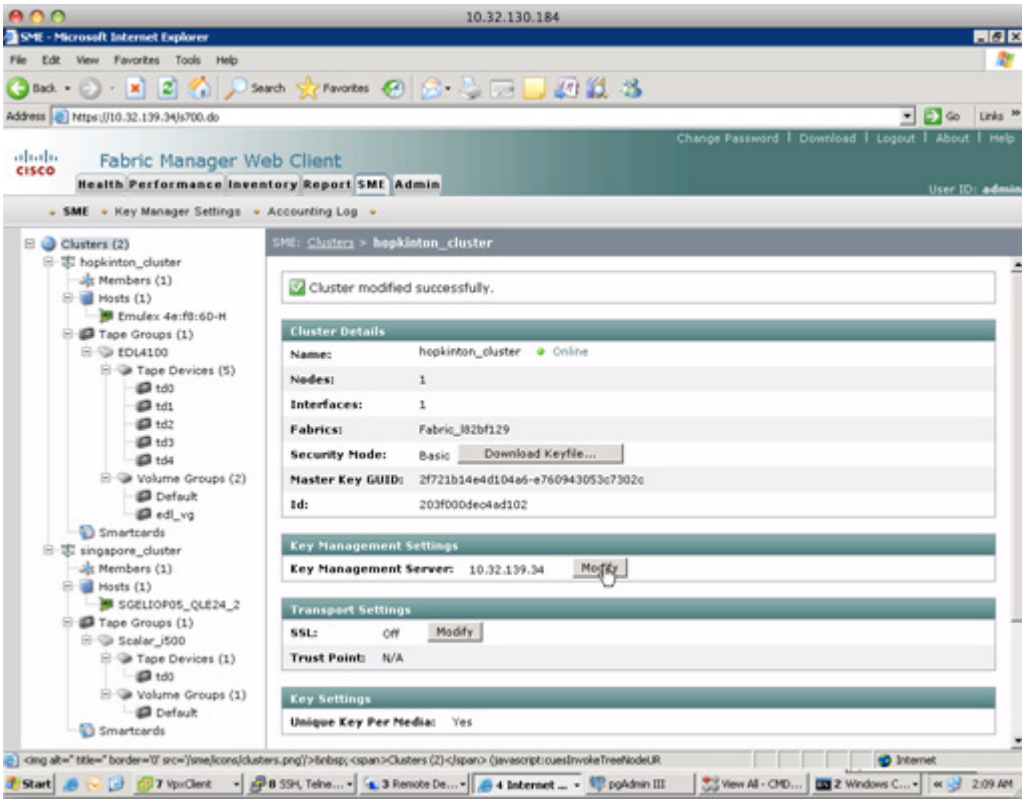
4. Change the KMC IP address on all sme clusters to point to KMC-B.

Note: All sme clusters now need to communicate using KMC-B for any key transactions. This will ensure that all key transactions are directed through the active KMC (KMC-B).

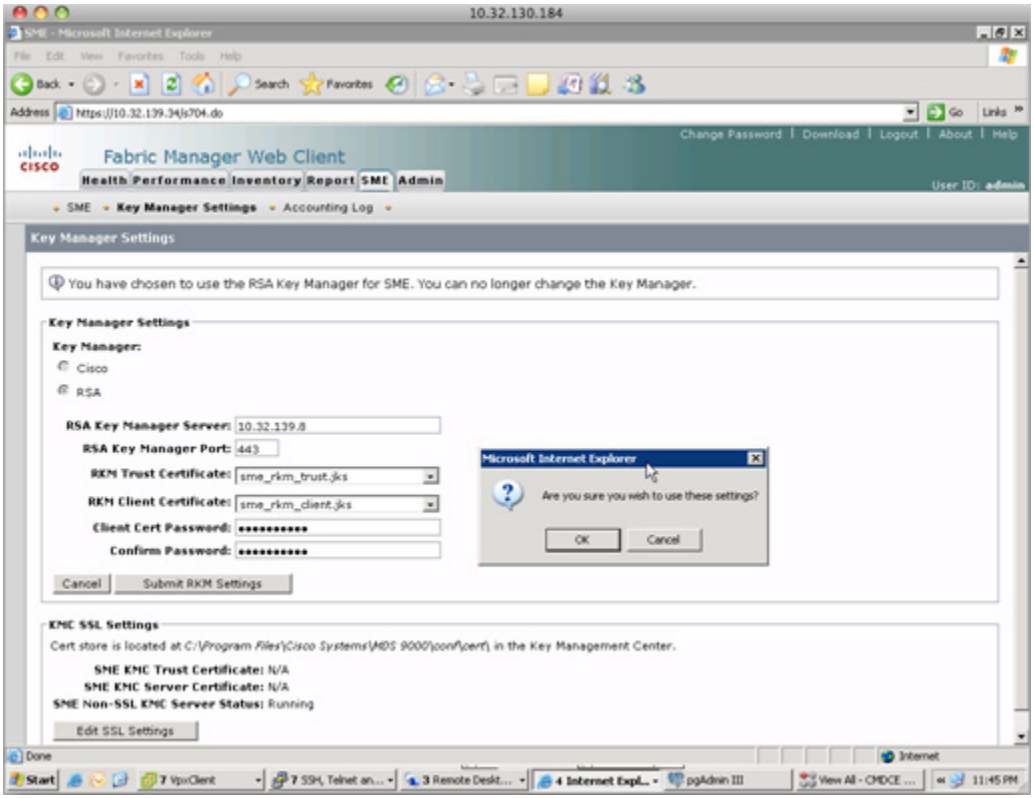
Do not restart KMC-A until all sme clusters point to KMC-B. This will prevent clusters from storing new information on a standby KMC (KMC-A).

IMPORTANT

This change cannot be performed on an archived cluster. You must change this information once the affected cluster is back in operation.

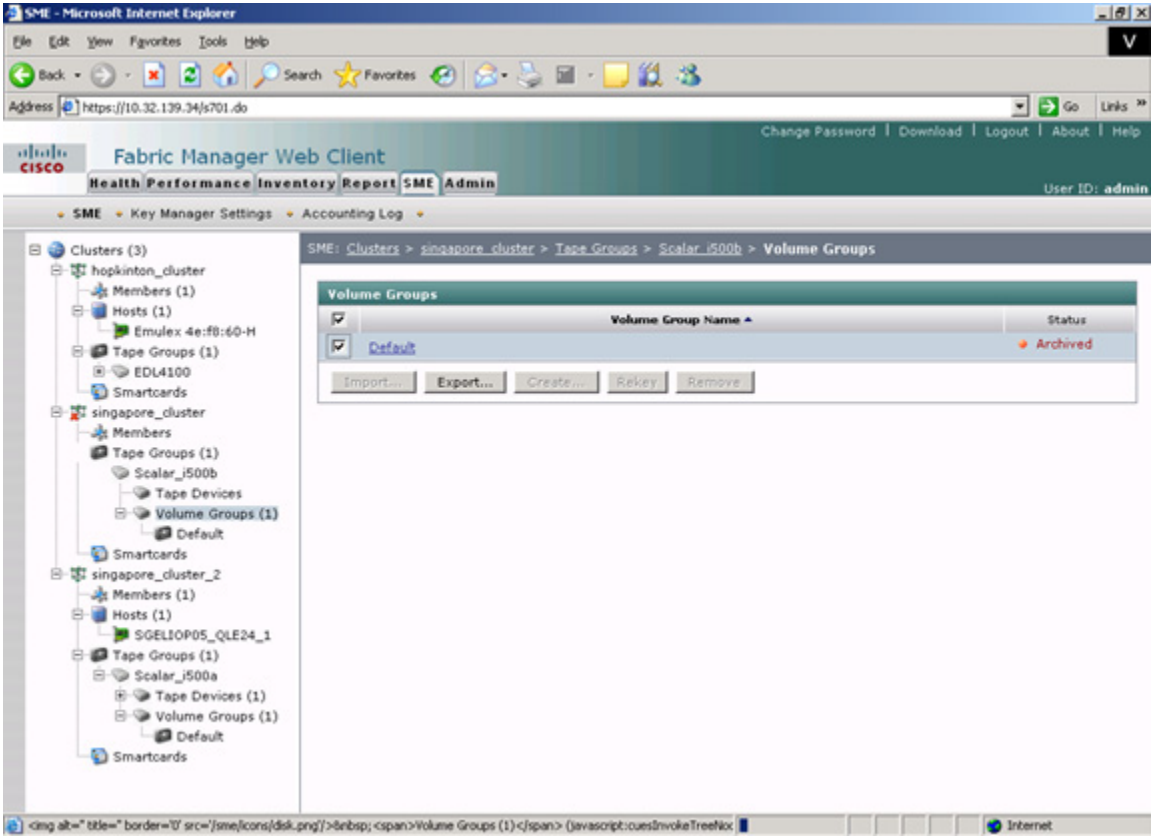


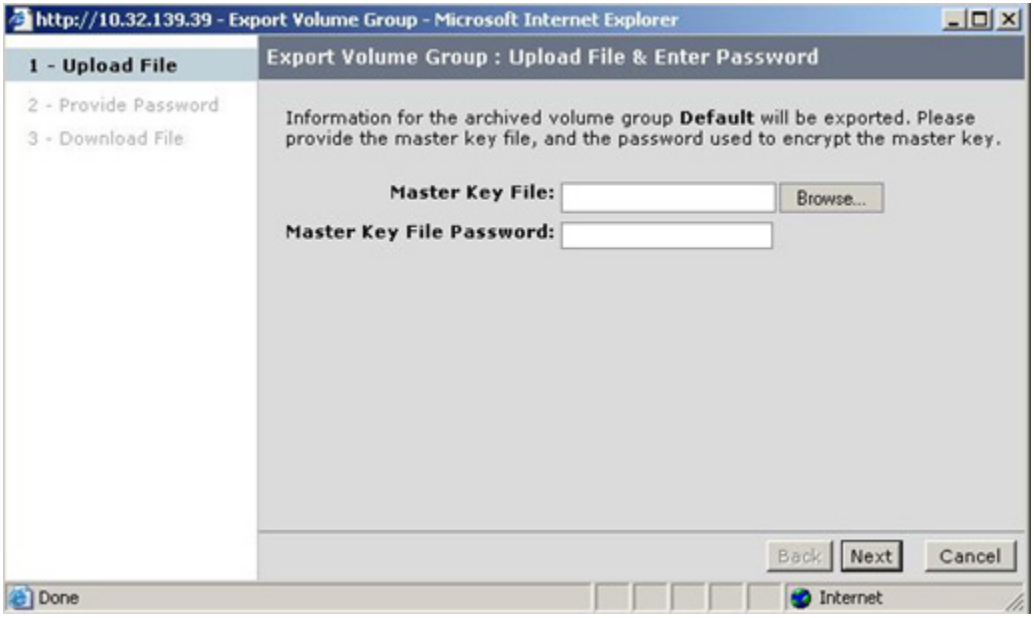
5. Change the RKM server setting in KMC-B to point to F5-B.
   KMC -B is now the active KMC.


10.32.130.184
Fabric Manager Web Client
Health Performance Inventory Report SME Admin
SME Key Manager Settings Accounting Log
Key Manager Settings
You have chosen to use the RSA Key Manager for SME. You can no longer change the Key Manager.
Key Manager Settings
Key Manager:
Cisco
RSA
RSA Key Manager Server: 10.32.139.8
RSA Key Manager Port: 443
RKM Trust Certificate: sme_rkm_trust.jks
RKM Client Certificate: sme_rkm_client.jks
Client Cert Password:
Confirm Password:
Cancel
Submit RKM Settings
Microsoft Internet Explorer
Are you sure you wish to use these settings?
OK
Cancel
KMC SSL Settings
Cert store is located at C:\Program Files\Cisco Systems\MDS 9000\conf\cert\ in the Key Management Center.
SME KMC Trust Certificate: N/A
SME KMC Server Certificate: N/A
SME Non-SSL KMC Server Status: Running
Edit SSL Settings

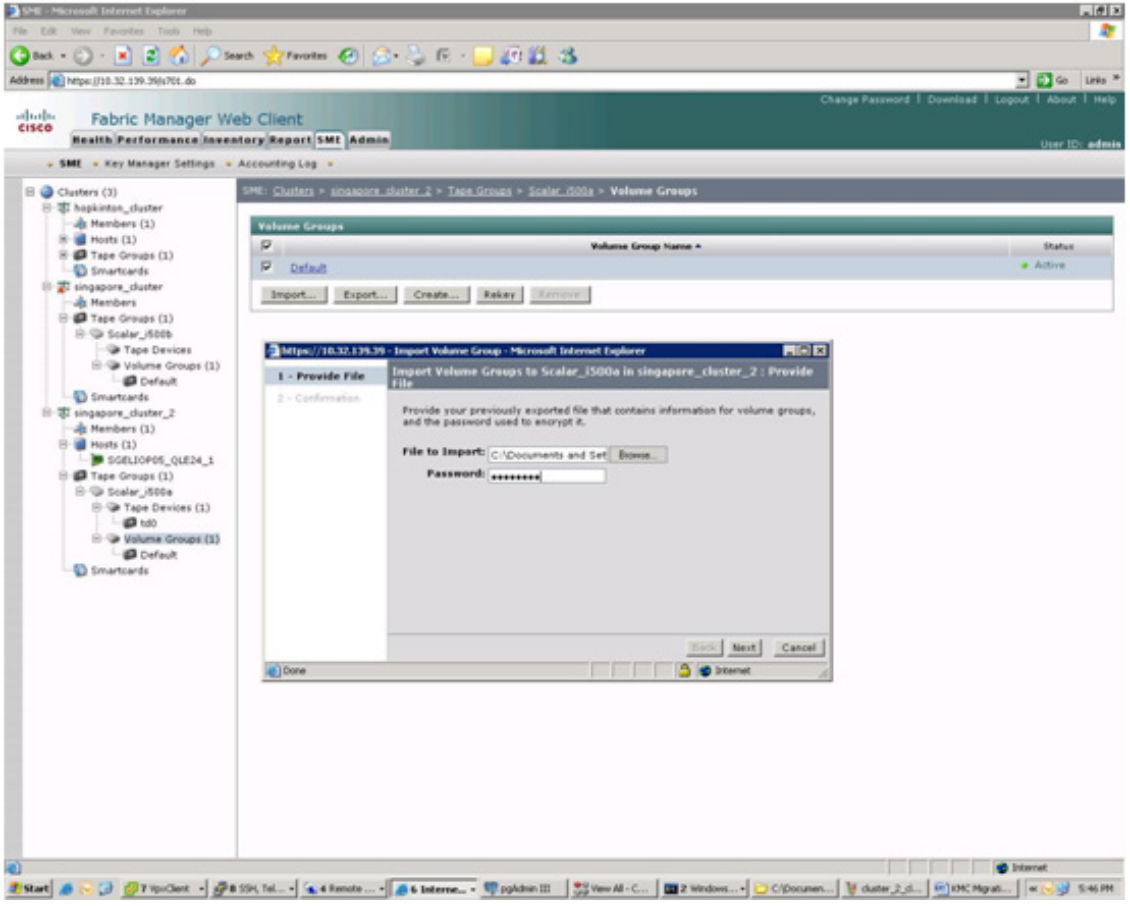
6. Perform an offline export of volume group key(s) that is part of the archived cluster.

Note: Depending on the security settings configured, this operation requires the smart card or the password to access the master key file for the archived cluster.




http://10.32.139.39 - Export Volume Group - Microsoft Internet Explorer
1 - Upload File
2 - Provide Password
3 - Download File
Export Volume Group : Upload File & Enter Password
Information for the archived volume group Default will be exported. Please provide the master key file, and the password used to encrypt the master key.
Master Key File:
Browse...
Master Key File Password:
Back
Next
Cancel
Done
Internet

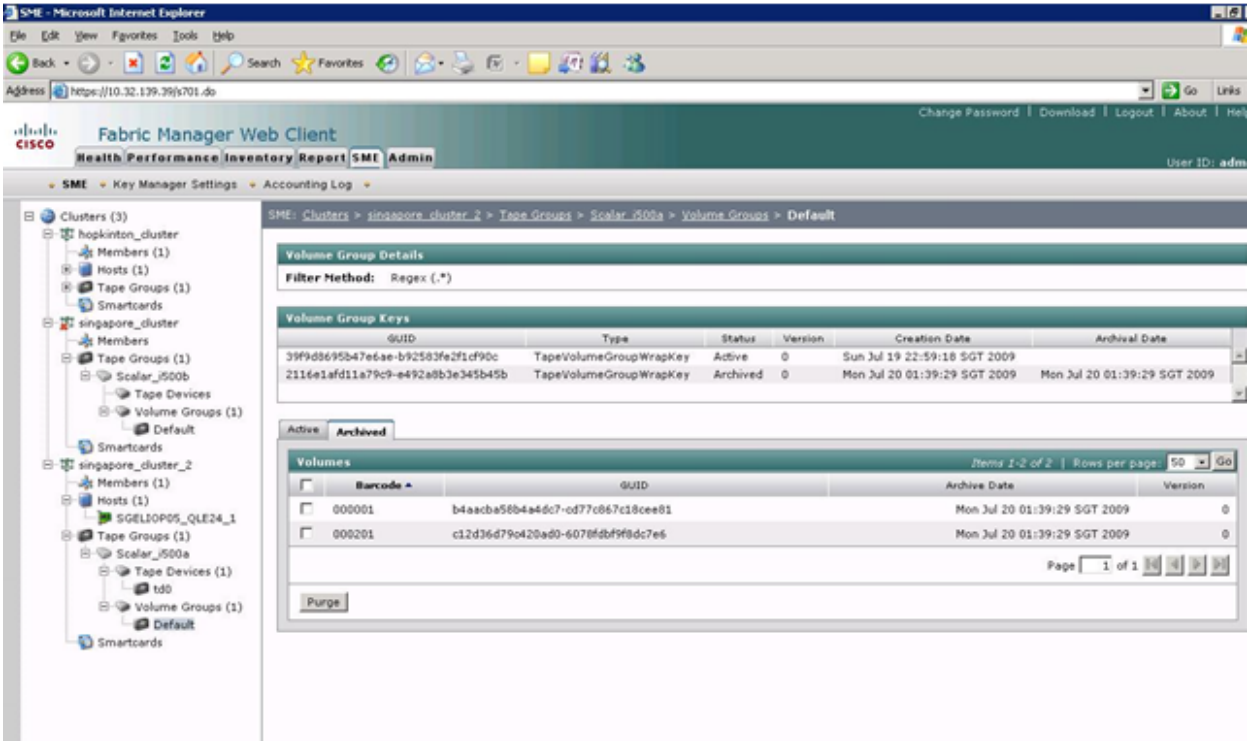
7. Import the exported key file into an existing online cluster.



8. Verify that the volume group key was successfully imported.

IMPORTANT

The imported volume groups will be under the Archived tab on the SME Volume group page.



KMC-B is now capable of restoring any tapes that were backed up via the affected cluster. The database on KMC-B must be backed up on a periodic basis to capture any changes made to the configuration.

IMPORTANT

In the event that you plan to migrate back to KMC-A, the latest backup of the database from KMC-B *must* be restored back to KMC-A as a part of the migration procedure that is detailed in "Step 1: Migration from two unique KMCs" on page 127.

7

Brocade Encryption Switch/Blade

This chapter provides information on implementing the fabric-based EMC/Brocade encryption solution.

- Introduction 152
- Fabric-based encryption solution terms and concepts 153
- Encryption topology basics 160
- Brocade fabric-based encryption case study 164
- TimeFinder case study 221
- SRDF encryption case study 232
- RecoverPoint encryption case study 293

Introduction

The EMC/Brocade encryption solution is fabric-based and provides IEEE 1619 compliant block-level encryption for the following:

- Disk storage based on the industry-standard AES-256-XTS encryption algorithm
- Tape storage based on the industry-standard AES-256-GCM encryption algorithm
- Virtual tape library support for the EMC EDL 4000 series
- Celerra NAS NDMP backup to physical tape

This encryption solution is based in part on the Fabric OS (FOS) embedded Name Server Frame Redirection (FR) feature, which directs traffic identified for encryption through Brocade's crypto-modules, referred to as E*ncryption Engines*. The use of Frame Redirection allows the crypto-modules to physically reside anywhere within a customer's FC SAN, alleviating the requirement of having to directly connect devices in the encryption path to a crypto-module.

Fabric-based encryption solution terms and concepts

This section briefly defines solution terms and concepts used for the EMC/Brocade fabric-based encryption solution.

FIPS and CC-EAL2 compliance

Brocade Encryption Engine (BEE) has been validated to comply with the Federal Information Processing Standard (FIPS) 140-2, Level 3 Security Requirements for Cryptographic Modules.

Data Encryption Key (DEK)

A Data Encryption Key (DEK) is an encryption key generated by the Encryption Engine. Every LUN configured for encryption is assigned its own unique DEK. The DEK is used to encrypt cleartext (readable) data before it is written to the LUN. Alternatively, it is also used to decrypt ciphertext (encrypted) data when it is read from the LUN.

Note: Metadata, known as the Key ID to identify the DEK, is written to the LUN. The DEK is then stored to a key vault, such as the RSA Key Manager (RKM) before encryption services are enabled for that LUN.

First-Time Encryption (FTE) or first-time rekeying

First-Time Encryption (FTE) is the encryption of existing cleartext data. In a first-time encryption operation, cleartext data is read from a LUN, encrypted with the current DEK, and written back to the LUN at the same Logical Block Address (LBA) location. This process, known as *in-place encryption*, effectively encrypts the LUN.

Online rekeying

In an online rekeying operation, while undergoing active IO, previously encrypted data on a LUN can be decrypted with the current DEK, re-encrypted with a new DEK, and then written back to the same LUN at the same LBA location. This process, known as in-place rekeying, re-encrypts the LUN with a new DEK.

Rekeying is always performed based on the security policies of an organization on key expiration. As a best practice, rekeying should be done only when necessary, since it can impact I/O performance to a LUN. The rekeying process should be limited to the following situations:

- When required by security policies, as infrequently as once a year.
- When the Master Key has been compromised.
- When the DEK has been compromised.
- When security has been breached by a trusted insider.

Rekeying is only applicable to disk array LUNs or fixed-block devices. Rekeying is *not* supported for tape media.

Active Master Key

The Active Master Key is used to encrypt and decrypt DEKs when storing DEKs to RKM key vaults. There is one master key per encryption group. That means all node encryption engines within an encryption group use the same master key to encrypt and decrypt the DEKs. You can restore the active Master Key under the following conditions:

- The active Master Key has been lost, which happens if all EEs in the group have been zeroized or replaced with new hardware at the same time.
- You want multiple encryption groups to share the same active Master Key, which might be the case when setting up multiple sites for replication environments, such as SRDF® or RecoverPoint.

Alternate Master Key

The alternate Master Key is used to decrypt DEKs that were not encrypted with the active Master Key. Restore the alternate Master Key for the following reasons:

- To read an old tape that was created when the group used a different active Master Key.
- To read a tape (or disk) from a different encryption group that uses a different active Master Key.

Encryption Engine (EE)

The Encryption Engine (EE) performs encryption operations, including the generation of the DEKs. In the EMC/Brocade encryption solution, the EE is an integral part of a Brocade Encryption Switch or a PB-DCX-16EB encryption blade, shown in Figure 21 on page 155.

Figure 21 **Brocade Encryption Switch (left) and PB-DCX-16EB encryption blade (right)**

Performance licensing

A performance license increases the encryption processing power of an EE. Purchasing the license makes the base solution of 48 Gigabits per second (Gb/s) scalable to 96 Gb/s.

Note: When installed on an ED-DCX-B Backbone, the license applies to all PB-DCX-16EB blades in that chassis.

Encryption node

A Brocade Encryption Switch represents a single encryption node and one EE. An ED-DCX-B Backbone with up to four PB-DCX-16EB blades also represents a single encryption node with up to four EEs.

An encryption node is identified by the WWN of the Brocade Encryption Switch or the ED-DCX-B in which PB-DCX-16EB blades are installed.

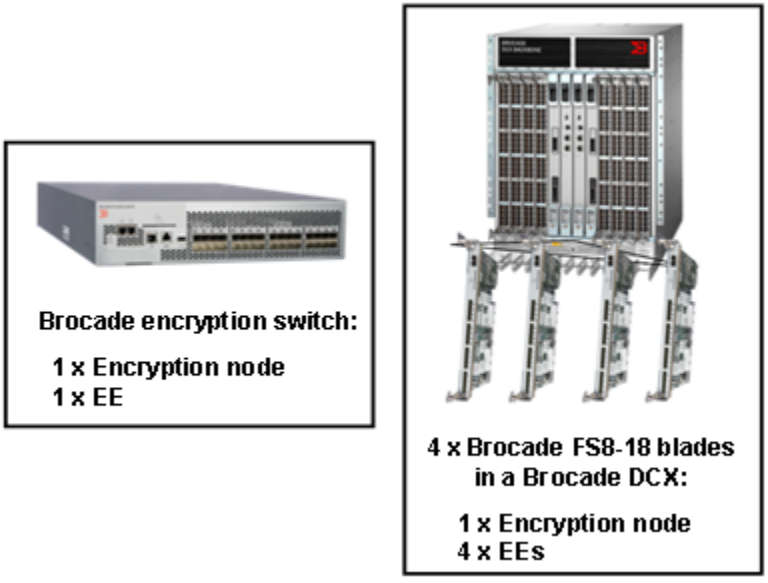


Figure 22 **Encryption nodes and EE**

Frame Redirection

Frame Redirection creates an abstraction layer (Redirection Zones) in the fabric, allowing frames to be redirected to the EE without modifying the existing zoning configuration. The abstraction layer creates a Virtual Initiator (VI) and a Virtual Target (VT). Frame Redirection still allows the existing zoned initiator and target to see

the World Wide Port Name (WWPN) and World Wide Node Names (WWNNs) of one another. However, the Port Identifiers (PIDs) are changed so the initiator sees the PID of the VT while the target sees the PID of the VI. As a result, frames are redirected from physical initiator to VT and physical target to VI through the crypto-module EE, as shown in Figure 23 on page 156.

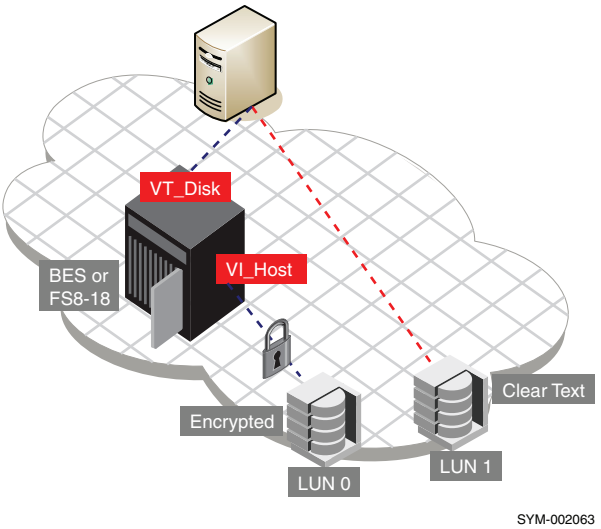


Figure 23 **Frame redirection through the encryption blade**

Data Encryption Key cluster

A DEK cluster is a group of EEs that provide access to the same LUN(s) within or across fabrics.

High Availability cluster

A High Availability (HA) cluster is a peer relationship between two EEs providing Crypto Target Container (explained in more detail on "Encryption node" on page 155) failover within a fabric, as shown in Figure 24.

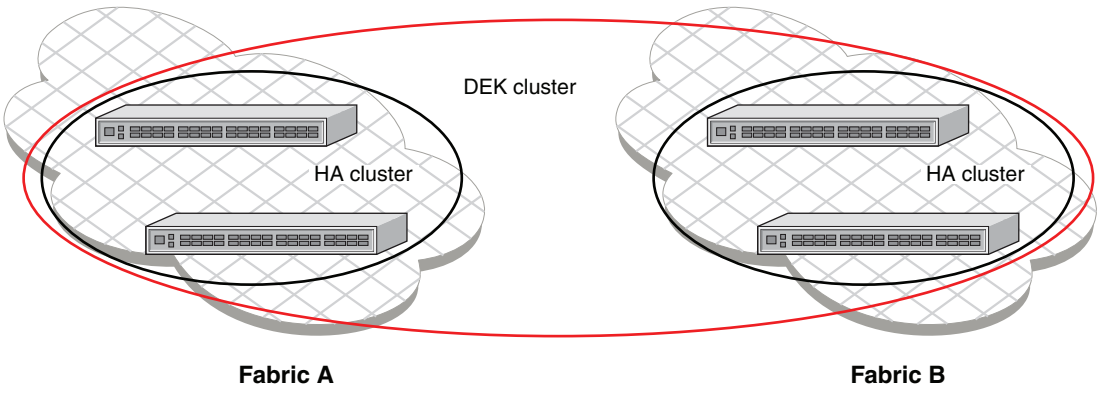


Figure 24 **HA clusters in a DEK cluster**

Encryption Group (EG)

An Encryption Group (EG) is a collection of one or more encryption nodes that share the same key vaults. The nodes can be inside or across fabrics. The nodes in an EG share the same DEKs and can provide access to the same LUNs on separate target ports enabling multipath access and failover capabilities.

Encryption Groups can consist of up to a maximum of four encryption nodes. If the four encryption nodes were ED-DCX-Bs, each could house up to four encryption engines (PB-DCX-16EB blades) resulting in the maximum encryption group supported configuration of sixteen encryption engines. If required, more than one EG can be set up within the same physical set of FC fabrics.

Crypto Target Container (CTC)

A Crypto Target Container (CTC) is defined for every storage port that will provide access to LUNs that require encrypting.

For disk LUNs, within each CTC, each disk LUN is defined and access to each LUN is set up for the appropriate hosts/initiators.

For tape media, within each CTC, tape drive LUNs are defined and access to each tape drive is setup for the appropriate hosts/initiators.

Upon creation of a CTC, Frame Redirection is enforced for the specified initiators and target port.

Note: A standard zone set between the initiator/target pair must be created prior to creating the CTC. Until this standard initiator/target zone has been created, the EE will not be allowed to access the CTC's Target port.

Once CTCs get created and committed to the EG configuration, frame redirection will immediately begin to take place and access to your LUNs/tapes will be precluded until/unless they are added to the CTC configuration.therefore, for online deployments, it is necessary to add the LUNs to the CTCs *prior* to committing the CTC configuration to the EG configuration.

The following objects make up a CTC, only one target port and one or more initiators:

CTC	Name
EE	WWNN (slot # if it is a blade)
Target	WWPN
Target	WWNN
Initiator	WWPN
Initiator	WWNN

Key vault

The key vault is a device, such as a RSA Key Manager (RKM), that provides key management functionality. Prior to the use of a DEK created by an EE, it must be backed up to the key vault.

Preemptive security breach features

Zeroizing

Zeroizing is the process of erasing all existing DEKs and other sensitive encryption information in an EE. Zeroizing has the following effects:

- The most important result is the complete secure wipe of the critical security parameters (CSP) essential for encryption, by design and conforming to FIPS 140-2 Level 3.
- All copies of data encryption keys kept in the encryption switch or encryption blade are erased.

  Note: Individual keys cannot be deleted in this manner.

- Internal public and private key pairs that identify the encryption engine are erased and the encryption switch or the encryption blade is put into the faulty state.
- All encryption operations on this engine are stopped and all VIs and VTs are removed from the fabric's name service.
- The Master Key is erased from the encryption engine. Once enabled, the encryption engine can restore the necessary DEKs from the key vault when the Master Key is restored.

- If the encryption engine was part of an HA cluster, targets fail over to the peer which assumes the encryption of all storage targets. Data flow will continue to be encrypted.
- If there is no HA backup, host traffic to the target will fail as if the target had gone offline. The host will not have unencrypted access to the target. No data flows, since the encryption VTs are offline.

Intrusion and tampering detection

The crypto module on the Brocade Encryption Switch and PB-DCX-16EB is located under a titanium cover and safeguarded by micro switches that detect intrusion and tampering. If intrusion is detected, tamper detection circuitry will trigger a zeroization of all DEKs and sensitive encryption information on the EE.

Note: After an EE has been zeroized, the EE is placed in a faulty state for increased protection. The faulty state can be recovered only by power cycling the Brocade Encryption Switch or performing **slotPowerOff/On** on the encryption blade.

Note: Zeroizing an EE affects I/O, but all target and LUN configuration remains intact.

Encryption topology basics

The following information is provided in this section.

- "Estimating the number of Encryption Engines needed" on page 160
- "Determining the placement of Encryption Engines" on page 161
- "Firmware level" on page 161
- "LAN assessment" on page 162
- "RSA Key Manager (RKM) key vaults" on page 163

Estimating the number of Encryption Engines needed

The sizing of an encryption solution would typically be performed as a joint effort between customers and sales personnel. Each encryption engine provides in-line encryption services with up to 96 Gb/s throughput for disk I/O (mix of ciphertext and cleartext traffic) and up to 48 Gb/s throughput for tape I/O (mix of ciphertext and cleartext traffic).

As an example, assume a customer had encryption requirements for a total of sixteen 4 Gb/s storage ports, eight from each fabric, resulting in a total of 64 Gb/s, or 32 Gb/s per fabric. Assuming that the storage arrays are delivering at line rate, 32 Gb/s of encryption processing power is needed per fabric at the time of implementation. Based on these calculations, a minimum of 1 x EE (48 Gb/s at base performance) per fabric would more than fulfill bandwidth requirements.

To meet the design principle of providing a scalable and highly available solution, a customer could consider implementing a total of 4 x EEs (2 EEs per fabric) incorporating HA clustering in each fabric. This configuration provides a total of 192 Gb/s (4 Brocade encryption switches x 48 Gb/s, or 96 Gb/s per fabric) of base encryption processing power. The proposed solution has a surplus of 128 Gb/s, allowing for approximately 200% growth.

If upgraded with an encryption performance license, the total encryption processing power is doubled to 384 Gb/s (192 Gb/s per fabric). Designing a base solution with a surplus of encryption processing power not only makes the design scalable but also

minimizes the performance impact on production in the event that an EE should become unavailable.

If instead of utilizing Brocade encryption switches, the customer opted to implement encryption blades installed in a ED-DCX-B chassis, processing power can continue to be increased. Up to four EEs can be installed per ED-DCX-B as long as there are available slots in the chassis. By increasing the number of EEs per fabric up to 4, the potential encryption processing power increases to 768 Gb/s (384 Gb/s per fabric) when the performance license is installed.

Note: Adding one more ED-DCX-B chassis per fabric with up to four PB-DCX-16EB blades doubles the processing power to a total of 1.5 Tb/s.

Determining the placement of Encryption Engines

Some care must be taken when connecting the encryption engines to the fabric. Although there is considerable flexibility in connecting and configuring the containers for encryption, the following guidelines are the recommended best practices:

- Host and storage array ports that are *not* involved in any encryption flow can be connected to any EEs.
- For high availability purposes, host and storage array ports that are involved in encryption flows should *not* be directly connected to any EEs.

Firmware level

To support Frame Redirection, the edge switches must be running FOS 6.1.0e or later and the ED-DCX-Bs must be running FOS 6.2.0g or later. In a fabric running Connectrix Enterprise OS (EOS), the minimum firmware level supported to interoperate with FOS 6.2.0g is EOS 9.8.0.

Note: Always consult the latest version of the *Connectrix B Fabric OS Release Notes* for supported firmware versions prior to upgrading.

LAN assessment

For communication between the encryption nodes and the key vaults, the management LAN interface is used. Therefore, you should assess the robustness of the management LAN to ensure that communication between encryption nodes and the key vaults is not hindered. This is important since DEK exchanges are performed between the encryption nodes and RKM key vaults on the management LAN interface.

For communication between the EEs, there are 2 x Gigabit Ethernet (GbE) ports (Ge0 and Ge1), also known as *IO synchronization links*. These links should be connected to a subnet or private LAN allowing EE-to-EE communication to be isolated from other traffic in the enterprise. Figure 25 on page 163 illustrates the LAN configuration.

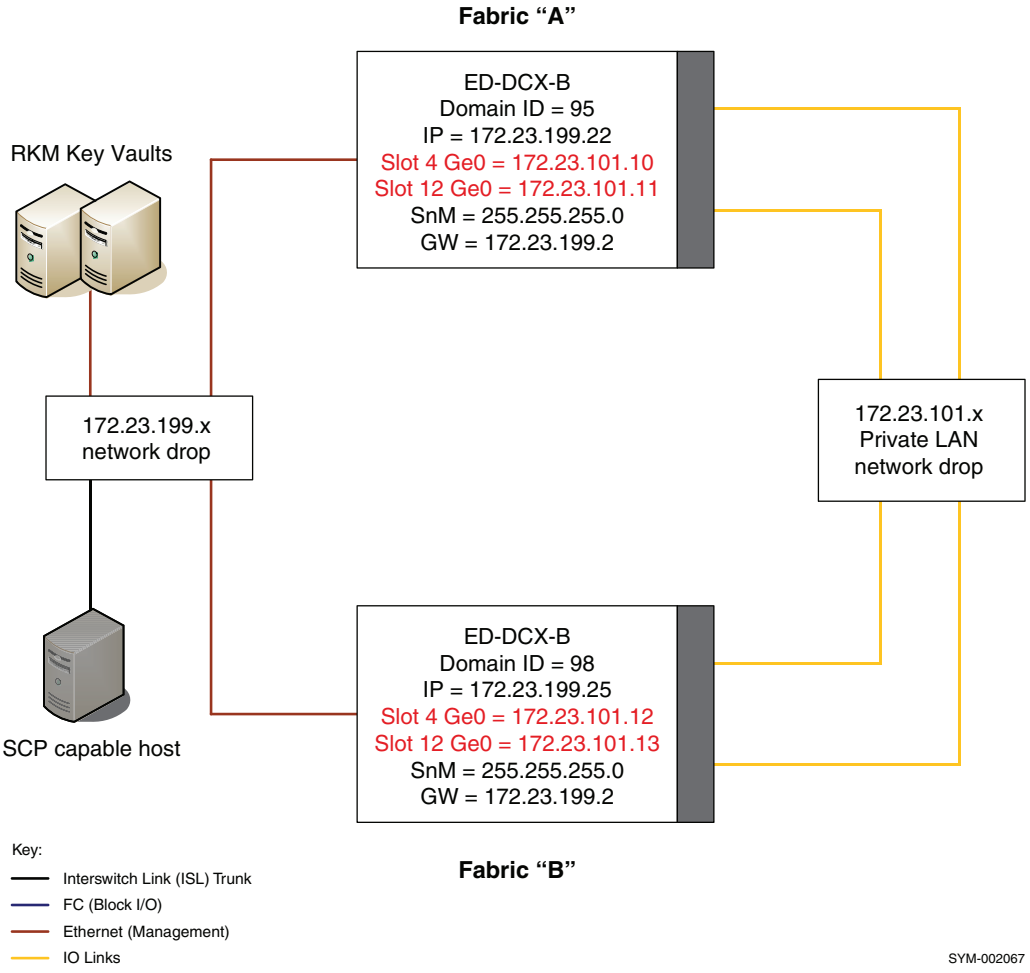


Figure 25 **LAN communication between Fabric A and Fabric B**

RSA Key Manager (RKM) key vaults

The RKM key vault appliance is preconfigured for central management of encryption keys throughout the enterprise. For redundancy, RKM key vault appliances are clustered together and are accessed via IP Load Balancers (IPLBs). Refer to the *EMC Support Matrix* (ESM) for a complete listing of supported RKM KV, FOS and IPLB software versions.

Brocade fabric-based encryption case study

This case study, based on FOS v6.2.x, describes how to design and deploy a Brocade fabric-based encryption. In the reference architecture shown in Figure 26 on page 165, the dual-fabric, core-edge design shows the ED-DCX-B at the core. The initiators are located at the edge and targets at the core.

This section contains the following information:

- "Important notes" on page 164
- "Target topology" on page 165
- "Summary of installation steps" on page 165
- "Summary of initialization steps" on page 168
- "Summary of CTCs, hosts and LUNs" on page 220

Important notes

Consider the following important notes concerning this case study:

- This encryption case study is based on FOS v6.2.x. "SRDF encryption case study" on page 232 is based on FOS v6.4.1x.
- There are minor command line output differences between the two FOS versions. Refer to the "SRDF encryption case study" on page 232, which is based on FOS v6.4.1a, for the most up-to-date command line outputs.
- The major difference between FOS v6.2.x and FOS v6.4.x implementations is how the RKM key vaults (KVs) are configured. In FOS v6.2.x, two standalone KVs were required; that is, the two KVs were not clustered in any way. As of FOS v6.3.x, two RKM KVs are required to be clustered and placed behind IP Load Balancers (IPLBs) to ensure high availability.
- The ability to utilize EMC TimeFinder, RecoverPoint, or SRDF was not qualified by EMC until FOS v6.4.x.

Target topology

Figure 26 shows the target topology for this case study.

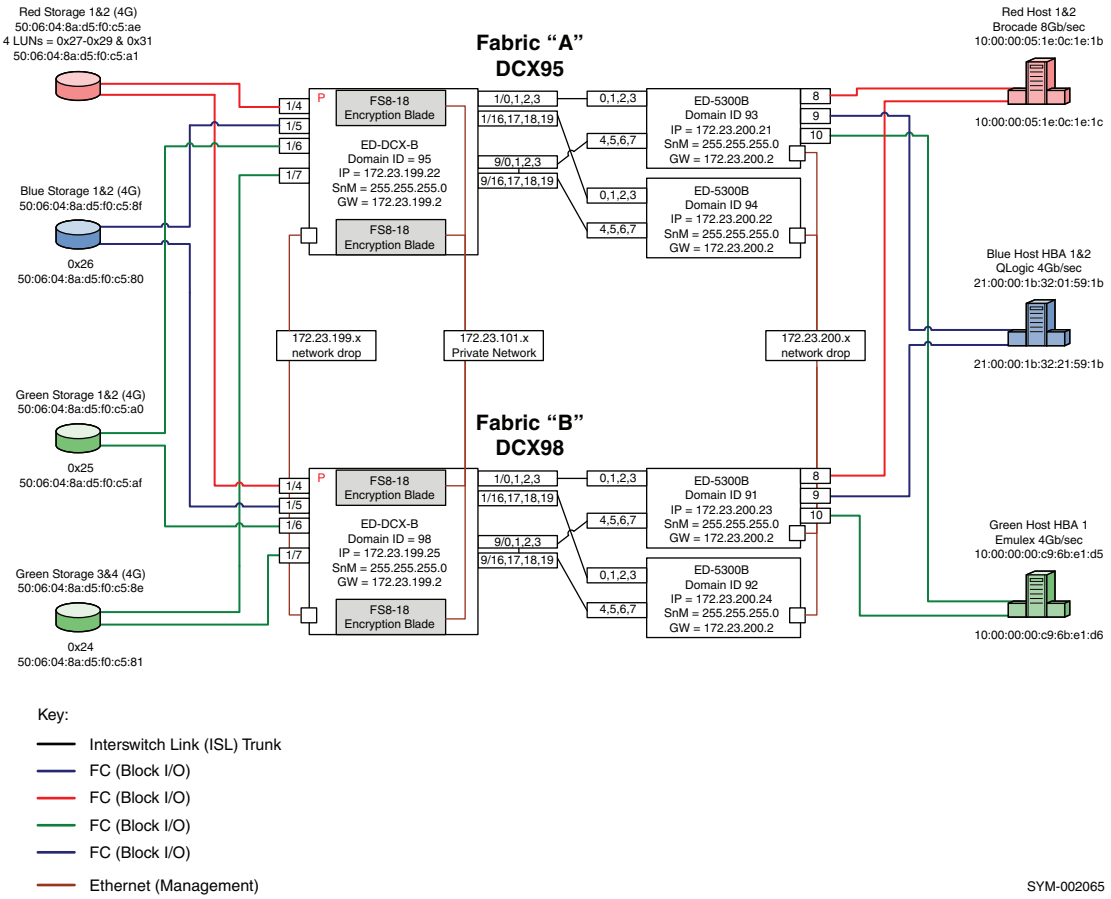


Figure 26 Dual-fabric, core-edge Storage Area Network (SAN), Target topology

Summary of installation steps

The following tasks are required for implementing fabric-based encryption. Each task is further described in this section:

- "Step 1: Configure the Brocade fabric " on page 166
- "Step 2: Plan the encryption solution" on page 166

- "Step 3: Plan the encryption management" on page 166
- "Step 4: Place EEs in the fabric" on page 167
- "Step 5: Plan key vault configuration and administration" on page 167
- "Step 6: Consider multipath access and failover" on page 167
- "Step 7: Create CTCs" on page 168

Note: The reference architecture uses the PB-DCX-16EB encryption blades for the ED-DCX-B as opposed to the standalone Brocade Encryption Switch.

Step 1: Configure the Brocade fabric

In order to configure the Brocade fabric, identify the following:

- The data, (files, applications, and so on) that need to be encrypted
- The number of initiators requiring encryption services
- The target ports correlated to the identified initiators
- The LUNs correlated to the initiators and targets
- The number of paths used for accessing the LUNs for initiators and targets

Note: CTCs can be comprised of a mixture of both encrypted and cleartext LUNs. Cleartext I/O flows move through the encryption engines and will therefore cut into the total available EE bandwidth. For example, if you had 48 Gb/s of traffic flows for cleartext LUNs added via CTCs, you would only have 48 Gb/s left for encrypted flows.

Step 2: Plan the encryption solution

Perform the following audits:

- Security audit of personnel such as administrators, managers, contractors, and so on
- SAN audit including physical location of equipment and existing zoning configuration
- Bandwidth requirement audit

Step 3: Plan the encryption management

Evaluate the following:

- The Command Line Interface (CLI) use
- Security Officers

- Graphical User Interface (GUI) tools, such as the Connectrix Manager Data Center Edition (recommended)

Step 4: Place EEs in the fabric

Consider the following:

- Placement of PB-DCX-16EB Encryption Blades in the ED-DCX-B at the core
- Bandwidth requirements to determine the number of EEs
- Attached devices in the fabric, such as existing storage

Step 5: Plan key vault configuration and administration

- Consider the number of key vaults needed at the time of implementation
- Evaluate key vault management
- Evaluate DEK life cycle management

Step 6: Consider multipath access and failover

- Evaluate the number of CTCs needed per fabric for multipath access
- Ensure that the existing multipath environment is in good working order
- Verify that all identified paths to a set of LUNs are available and standard failover and load balancing are working properly
- Identify the initiators that require encryption services and the number of paths to a specific set of LUNs

  For example: If an initiator has two paths to a set of three LUNs via one target port in both Fabrics A and B, this would result in the creation of a total two CTCs (one for each fabric). Consider the Red Host in Figure 26 on page 165. A CTC would be created for Red Storage 1 WWPN/WWNN in Fabric A containing the WWPN/WWNN of Red Host 1 followed by creation of CTC for Red Storage 2 WWPN/WWNN in Fabric B containing Red Host 2 WWPN/WWNN. More importantly, the same set of three LUN(s) would to be added to each of the CTCs.

  For this deployment, the reference architecture assumes that all data is to be encrypted. Since there are 8 physical storage ports, a total of 8 CTCs (4 per fabric) need to be created. Assigning the appropriate host and defining the LUNs for each CTC completes the task.

In summary, identifying the number of target port and paths for a specific set of LUNs ensures the success of the encryption solution

Step 7: Create CTCs

- Verify the WWNs of the encryption nodes and slot numbers for EEs to be used for both fabrics
- Gather the WWPN and WWNN of the initiator and target ports that require multipath access through CTCs
- Verify LUN numbers and LUN Serial Numbers (LSNs) to ensure multipath access between fabrics in the CTCs
- Evaluate the amount of data for First Time Encryption (FTE) once LUNs are placed in a CTC

Summary of initialization steps

Setting up Brocade fabric-based encryption requires initialization steps, which are performed only once and which must be executed in the specified order indicated.

The following is a high-level overview of the configuration process. Each step is explained in more detail in subsequent sections.

- "Step 1: Install encryption blades" on page 168
- "Step 2: Initialize Encryption Node DCX95 in Fabric A" on page 172
- "Step 3: Initialize Encryption Node DCX98 in Fabric B" on page 185
- "Step 4: Configure HA clusters for EEs on DCX95 and DCX98 in Fabrics A and B" on page 190
- "Step 5: Create CTCs in both Fabrics A and B" on page 192

Step 1: Install encryption blades

To install the encryption blades, complete Step 1 through Step 4:

1. Install the encryption blades in the ED-DCX-B

   Install a total of four PB-DCX-16EB Encryption Blades (two per ED-DCX-B) into empty slots 4 and 12 in the DCX95 for Fabric A and in the DCX98 for Fabric B. Then verify that the blades have successfully powered on using the **slotShow** command. When

the PB-DCX-16EB blades are initially inserted, it takes several minutes for them to become enabled, as shown in the following output in which "AP BLADE 43" is represented in slots 4 and 12.

```
DCX95:admin> slotshow

Slot   Blade Type      ID    Status
-----------------------------------
  1     SW BLADE      55     ENABLED
  2     UNKNOWN              VACANT
  3     UNKNOWN              VACANT
  4     AP BLADE      43     ENABLED
  5     CORE BLADE    52     ENABLED
  6     CP BLADE      50     ENABLED
  7     CP BLADE      50     ENABLED
  8     CORE BLADE    52     ENABLED
  9     SW BLADE      55     ENABLED
 10     UNKNOWN              VACANT
 11     UNKNOWN              VACANT
 12     AP BLADE      43     ENABLED

DCX98:admin> slotshow

Slot   Blade Type      ID    Status
-----------------------------------
  1     SW BLADE      55     ENABLED
  2     UNKNOWN              VACANT
  3     UNKNOWN              VACANT
  4     AP BLADE      43     ENABLED
  5     CORE BLADE    52     ENABLED
  6     CP BLADE      50     ENABLED
  7     CP BLADE      50     ENABLED
  8     CORE BLADE    52     ENABLED
  9     SW BLADE      55     ENABLED
 10     UNKNOWN              VACANT
 11     UNKNOWN              VACANT
 12     AP BLADE      43     ENABLED
```

2. Zeroize the EEs in each blade.

   Zeroize the EEs in each blade using the **cryptocfg -zeroizeEE <*slot*#>** command. Add the slot number for the slot in which the PB-DCX-16EB blade is installed:

```
DCX95:admin>cryptocfg -zeroizeEE 4
This will zeroize all critical security parameters
ARE YOU SURE (yes, y, no, n): [no]y
Operation succeeded

DCX95:admin>cryptocfg -zeroizeEE 12
This will zeroize all critical security parameters
ARE YOU SURE (yes, y, no, n): [no]y
```

```
Operation succeeded

DCX98:admin>cryptocfg -zeroizeEE 4
This will zeroize all critical security parameters
ARE YOU SURE (yes, y, no, n): [no]y
Operation succeeded

DCX98:admin>cryptocfg -zeroizeEE 12
This will zeroize all critical security parameters
ARE YOU SURE (yes, y, no, n): [no]y
Operation succeeded
```

The **zeroizeEE** command leaves the EE in a faulty state that requires the blade to be power cycled as follows:

```
DCX95:admin> slotshow

Slot   Blade Type      ID    Status
-----------------------------------
  1      SW BLADE      55     ENABLED
  2      UNKNOWN              VACANT
  3      UNKNOWN              VACANT
  4      AP BLADE      43     FAULTY (21)
  5      CORE BLADE    52     ENABLED
  6      CP BLADE      50     ENABLED
  7      CP BLADE      50     ENABLED
  8      CORE BLADE    52     ENABLED
  9      SW BLADE      55     ENABLED
 10      UNKNOWN              VACANT
 11      UNKNOWN              VACANT
 12      AP BLADE      43     FAULTY (21)

DCX98:admin> slotshow

Slot   Blade Type      ID    Status
-----------------------------------
  1      SW BLADE      55     ENABLED
  2      UNKNOWN              VACANT
  3      UNKNOWN              VACANT
  4      AP BLADE      43     FAULTY (21)
  5      CORE BLADE    52     ENABLED
  6      CP BLADE      50     ENABLED
  7      CP BLADE      50     ENABLED
  8      CORE BLADE    52     ENABLED
  9      SW BLADE      55     ENABLED
 10      UNKNOWN              VACANT
 11      UNKNOWN              VACANT
 12      AP BLADE      43     FAULTY (21)
```

3. Power cycle each blade:

```
DCX95:admin>slotpoweroff 4
```

```
DCX95:admin>slotpoweron 4

DCX95:admin>slotpoweroff 12
DCX95:admin>slotpoweron 12


DCX98:admin>slotpoweroff 4
DCX98:admin>slotpoweron 4

DCX98:admin>slotpoweroff 12
DCX98:admin>slotpoweron 12
```

4. Ensure that the blades are enabled.

   After zeroizing and power cycling, ensure that the blades are once again enabled. It takes several minutes for the blade to become enabled after the power cycle.

```
DCX95:admin> slotshow

Slot   Blade Type      ID    Status
----------------------------------
  1      SW BLADE      55     ENABLED
  2      UNKNOWN              VACANT
  3      UNKNOWN              VACANT
  4      AP BLADE      43     ENABLED
  5      CORE BLADE    52     ENABLED
  6      CP BLADE      50     ENABLED
  7      CP BLADE      50     ENABLED
  8      CORE BLADE    52     ENABLED
  9      SW BLADE      55     ENABLED
 10      UNKNOWN              VACANT
 11      UNKNOWN              VACANT
 12      AP BLADE      43     ENABLED

DCX98:admin> slotshow

Slot   Blade Type      ID    Status
----------------------------------
  1      SW BLADE      55     ENABLED
  2      UNKNOWN              VACANT
  3      UNKNOWN              VACANT
  4      AP BLADE      43     ENABLED
  5      CORE BLADE    52     ENABLED
  6      CP BLADE      50     ENABLED
  7      CP BLADE      50     ENABLED
  8      CORE BLADE    52     ENABLED
  9      SW BLADE      55     ENABLED
 10      UNKNOWN              VACANT
 11      UNKNOWN              VACANT
 12      AP BLADE      43     ENABLED
```

Step 1 is now complete and the following checkpoint reached.

CHECKPOINT:

All PB-DCX-16EB encryption blades in both ED-DCX-B Backbones in Fabrics A and B are ready for configuration with the RKM key vaults.

Step 2: Initialize Encryption Node DCX95 in Fabric A

Note: A group leader is the first node created in an encryption group that acts as a group and cluster manager. The group leader manages and distributes all group-wide and cluster-wide configurations to all members of the group or cluster. If the group leader node is power cycled, another group member becomes the group leader. When the previous group leader comes back online, it becomes a group member.

To initialize Encryption Node DCX95 in Fabric A and to complete the initial setup of the RKM, complete Step 1 through Step 27:

1. Generate security parameters and certificates.

   Use the **initNode** command, which generates the following security parameters and certificates:

   - Node CP certificate
     - This is created during node initialization (c**ryptocfg --initnode**), exchanged with the group leader, and used for authenticating a member node with the group leader.
   - Authentication Center (KAC) certificate *or* CSR (Certificate Sign and Request)
     - The KAC certificate is a self-signed certificate would could be used for authentication with the RSA Key Manager (RKM) key vault
     - The CSR is a KAC signing request certificate which would need to be signed by a Certificate Signing Authority
   - FIPS Crypto Officer
     - This is used for internal authentication between processors. This certificate is exchanged internally and therefore requires no manual intervention.
   - FIPS user
     - Like the FIPS Crypto Officer, this certificate is also used for internal authentication between processors and is exchanged internally, so requires no manual intervention.

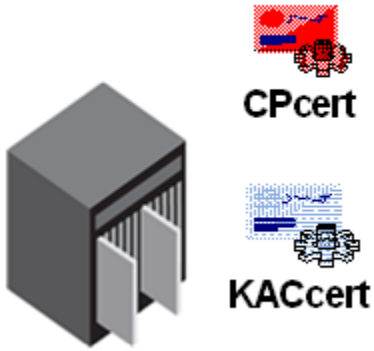


Figure 27 **Initnode command creates certificates for node to be shared or exported**

This command is used only once for each ED-DCX-B and must not be repeated if additional encryption blades are later added to the ED-DCX-B:

```
DCX95:admin>cryptocfg -initnode
This will overwrite all identification and authentication data
ARE YOU SURE (yes, y, no, n): [no] y
Notify SPM of Node Cfg
Operation succeeded.
```

2. Create an encryption group called "brocade."

   Command syntax:

```
DCX95:admin>cryptocfg --create -encgroup <encryption group name>
```

   Example:

```
DCX95:admin>cryptocfg -create -encgroup brocade
```

3. Initialize the EE.

   Initialize the EE to generate critical security parameters and certificates in the crypto module using the **initEE** command and specifying the slot:

```
DCX95:admin>cryptocfg -initEE 4
This will overwrite previously generated identification
and authentication data
ARE YOU SURE (yes, y, no, n): y
Operation succeeded.

DCX95:admin>cryptocfg -initEE 12
This will overwrite previously generated identification
```

```
and authentication data
ARE YOU SURE (yes, y, no, n): y
Operation succeeded.
```

4. Register the EE with the chassis using the **regEE** command and specifying the slot:

```
DCX95:admin>cryptocfg –regEE 4
Operation succeeded.

DCX95:admin>cryptocfg –regEE 12
Operation succeeded.
```

This step exchanges the FIPS Crypto officer and FIPS user certificates between the crypto module (responsible for all encryption and decryption operations) and the control processor. This is done for authentication purposes.

5. Set RKM as the key vault type.

   To set the key vault type for the entire encryption group, use the **cryptocfg --set –keyvault** command with the RKM option:

```
DCX95:admin>cryptocfg --set -keyvault RKM
Set key vault status: Operation Succeeded.
```

6. Request the DCX95 switch certificate signing, import the signed certificate, and register the certificate on the group leader.

   a. Export the KAC certificate signing request from the group leader node to an SCP-capable host.

      The command syntax is as follows:

```
DCX95:admin>cryptocfg --export -scp -KACcsr <IP address of SCP-capable server>
   <host_username> <host_filepath and filename>

DCX95:admin>cryptocfg --export -scp -KACcsr 172.23.199.75 admin
   /home/admin/certs/DCX95_kac_cert.pem
### Get FN </etc/fabos/certs/sw0/kac_req.csr> ###
admin1@172.23.199.75's password:
Operation succeeded.
```

   b. Submit the KAC certificate "DCX95_kac_cert.pem" to a Certificate Authority (CA) for signing.

Note: There are many third-party certification signing authorities (CAs) that can be used to sign your Certificate Signing Requests (CSRs). The process simply involves submitting the CSR to the CA such as VeriSign. The CA would then return the signed certificate via the Web, email, SCP, or other option depending on desired security.

---

c. Store the signed KAC certificate on the SCP-capable host. This certificate will be imported later into the EG leader node.

Note: You will need the store the Certificate Authorities (CAs) Trusted Root certificate on the SCP-capable host as well. his certificate will be supplied by the CA.

For the purposes of this example, it is assumed that the signed GL Node KAC certificate is called DCX95_kac_cert_signed.pem and stored on the SCP-capable host in the /home/admin/certs directory. It is further assumed that the CA Trusted Root certificate is called ABC_Signing_CA.pem and stored in the same /home/admin/certs directory.

Figure 28 **KAC signing and storage process**

d. The signed KAC certificate DCX95_signed_kac_cert.pem must now be placed in two different locations, as follows:

- On the RKM appliance (the RKM appliance is accessed via a Web Browser) associated with the RKM Identity for the DCX95 encryption group node (details are provided in Step 17 on page 179).
- Imported back onto DCX95 (details are provided in Step 18 on page 180) as shown Figure 29 on page 176.

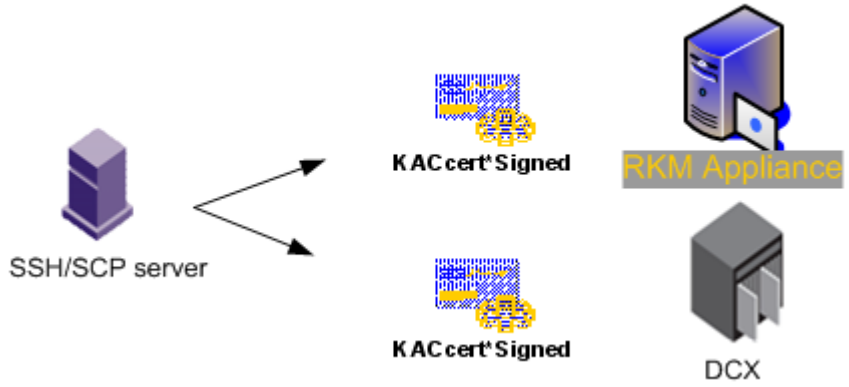


Figure 29 **Signed KAC certificate storage process**

To summarize our steps to this point, we have exported a KAC CSR to a CA to be signed. Subsequently, we've taken that signed KAC in addition to the CA's Trusted Root certificate and stored them on an SCP-capable server.

e. We must now take the RKM CA cert (subsequently referred to as RKM_cert.pem) and import it into the Group Leader node, as shown in Figure 30. (This certificate could technically be called anything and is created when the RKM software is initialized on the RKM appliance.) To import the RKM CA cert, you will first need to SCP the RKM_cert.pem file from the RKM appliance to an SCP-capable host. For this example, the RKM_cert.pem file will be placed in the SCP-capable host's /home/admin/certs directory.

Figure 30 **RKM Certificate transfer to Group Leader node process**

f. Import the signed certificate file from the SCP-capable host.

– Use the **cryptoCfg** command to import a file from SCP capable host.
– Use the **crytpocfg command** to import the signed KAC file from the SCP capable host.

Use the following command syntax:

```
DCX95:admin>cryptocfg --import -scp <local name> <host IP> <host username> <host
   path>
```

For example:

```
DCX95:admin>cryptocfg --import -scp DCX95_kac_cert_signed.pem 172.23.199.75 admin
   /home/admin/certs/DCX95_kac_cert_signed.pem
Available Space:16384
Make sure your file size is not greater than 16384.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to proceed?
ARE YOU SURE  (yes, y, no, n): [no] yes
admin1@172.23.199.75's password:
Operation succeeded.
```

g. Register the signed KAC certificate:

```
DCX95:admin>cryptocfg --reg -KACcert DCX95_kac_cert_signed.pem
kac.0000000051e460800
Register KAC status:
Operation Succeeded.
```

This first-time setup involves registering each encryption group node with the RKM key vault(s). Each encryption group node is assigned an identity on the RKM key vault and that identity is associated with the signed KAC certificate. In addition, the RKM setup procedure configures operating parameters for the encryption group environment such as Key Classes and Identity Groups.

7. Export the signed KAC certificate (**DCX95_kac_cert_signed.pem)** to the SCP-capable host.

```
DCX95:admin>cryptocfg --export -scp -KACcert DCX_kac_cert_signed.pem
   172.23.199.75 admin /home/admin/certs/DCX95_kac_cert_signed.pem
```

8. You will need a browser to browse to the RKM appliance. If your SCP-capable host does not have browsing capabilities, it will be necessary to transfer the KAC certificate (DCX95_kac_cert_signed.pem) and CA certificate (ABC_Signing_CA.pem) from the SCP-capable host to a workstation with browsing capabilities.

9. From the SCP capable host or workstation, open a Web browser and connect to the setup page using the URL https://rkm_server_network_name' (as in https://rkm_server_network_name/rkmawa). The rkm_server_network_name is the network name corresponding to the IP address of the RKM server. You also need the proper authority level, a username, and a password.
10. Select the **Operations** tab.
11. Select **Certificate Upload**.
12. In the **SSLCAcertificateFile field**, enter the full local path of the CA certificate (/home/admin/certs/RKM_cert.pem).

    IMPORTANT

    If the appliance is being shared, be sure to append the CA certificate to the existing uploaded certificate. You may inadvertently overwrite existing certificates if this is not done.

13. Select **Upload** >**Configure SSL > Restart Webserver**.
14. After the Web server restarts, enter the root password.
15. Open another Web browser and start the RKM management user interface.

    You will need the URL https://rkm_server_network_name' (as in https://rkm_server_network_name/KMS/login.jsp). The rkm_server_network_name is the network name corresponding to the IP address of the RKM server. You also need the proper authority level, a user name (kmsadmin), and a password.

    An Identity Group called *Hardware Retail Group* must be present on the RKM server. To create the Hardware Retail Group Identity Group, if it doesn't already exist, click the **Identity Group** tab, click **Create,** type **Hardware Retail Group** as the Identity Group name, and then select **OK**.
16. Select the **Key Classes** tab.

    For each of the following key classes, perform Step a through Step h to create the class. The key classes must be created only once, regardless of the number of nodes in your encryption group and regardless of the number of encryption groups that will be sharing this RKM.

    kcn.1998-01.com.brocade:DEK_AES_256_XTS

kcn.1998-01.com.brocade:DEK_AES_256_CCM

kcn.1998-01.com.brocade:DEK_AES_256_GCM

kcn.1998-01.com.brocade:DEK_AES_256_ECB

a. Click **Create**.

b. Type the key name string into the **Name** field.

c. Select **Hardware Retail Group** for **Identity Group**.

d. Deselect **Activated Keys Have Duration**.

e. Select **AES** for **Algorithm**.

f. Select **256** for **Key Size**.

g. Select the **Mode** for the respective key classes as follows:

- **XTS** for Key Class "kcn.1998-01.com.brocade:DEK_AES_256_XTS"
- **CBC** for Key Class "kcn.1998-01.com.brocade:DEK_AES_256_CCM"
- **CBC** for Key Class "kcn.1998-01.com.brocade:DEK_AES_256_GCM"
- **ECB** for Key Class "kcn.1998-01.com.brocade:DEK_AES_256_ECB"

h. Click **Next**.

i. Repeat the instructions in this step for each key class.

j. Click **Finish**.

Note: KAC certificates are listed as issued to and issued by kac.000000aabbccddee where "aabbccddee" are the last five bytes of the switch WWN. If the switch has been re-initialized, make sure to delete the previously imported certificate before using the new certificate. Both certificates have the same WWN but they have different creation dates.

---

17. In the RKM management GUI, create a new Identify for the Group Leader node as follows:

Note: Each Encryption Group Node must have an Identity created for it on the RKM KV. This allows the RKM to identify and authenticate the EG Node by its associated KAC certificate.

---

a. Click the **Identities** tab.

b. Click **Create**.

c. Enter the ED-DCX-B switch name into the **Name** field.

d. Select **Hardware Retail Group** as the Identity Group.

e. Select **Operational User** as the Role.

f. Click **Browse** and select the signed DCX95_kac_cert_signed.pem as the Identity Certificate.

g. Click **Save**.

18. Import the RKM certificate file (RKM_cert.pem) from the SCP-capable host onto the group leader.

Note: The following takes the RKM certificate from the SCP-capable host location (/home/admin/certs/RKM_ cert.pem) and imports the file to the group leader node. The RKM_ cert.pem had been placed onto the SCP-capable host in the location /home/admin/certs/ in Step 7.

```
DCX95:admin>cryptocfg --import -scp RKM_cert.pem 172.23.199.75 admin
   /home/admin/certs/RKM_ cert.pem
Available Space:16384
Make sure your file size is not greater than 16384.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to proceed?
ARE YOU SURE  (yes, y, no, n): [no] yes
admin1@172.23.199.75's password:
Operation succeeded.
```

19. Register RKM as the primary key vault for the group leader.

    Use the RKM CA certificate (RKM_cert.pem) that was imported to the DCX95 in the previous step.

    The command syntax is as follows:

```
DCX95:admin>cryptocfg --reg -keyvault <cert label> <certfile> <IP address>
<primary | secondary>
```

    For example:

```
DCX95:admin>cryptocfg --reg -keyvault RSA_cert RKM_cert.pem 172.23.199.75 primary
```

20. Generate and export the Master Key.

    A Master Key must be created on the group leader and exported to a backup location. This can be on the RKM key vault or an SCP-capable host.

```
DCX95:admin>cryptocfg --genmasterkey
```

Note: All DEKs stored in the RKM key vault are encrypted with a Master Key. This Master Key, once generated by the EG leader, must be backed up before the EG is allowed to perform data encryption operations.

There is an active Master Key and there can also be an alternate Master Key. Only the active Master Key can be backed up. It is very important to have a backup of the Master Key, because without it you cannot decrypt any DEKs retrieved from RKM key vaults and existing encrypted data can no longer be decrypted.

IMPORTANT

Be careful when choosing the method to store the Master Key. You can back up or restore the Master Key to the key vault, to a file, or to a set of smart cards. Using the smart card set is the most secure approach. There is no CLI option to back up the Master Key to a set of smart cards. This is only available through the GUI.

A user with malicious intent gaining access to the Master Key would conceivably be able to decrypt DEKs encrypted with the Master Key and then use those DEKs to decipher encrypted data.

21. Export the Master Key and choose a passphrase.

```
DCX95:admin>cryptocfg --exportmasterkey
Enter the passphrase: <INSERT PASSPHRASE HERE>
Master key exported. Key ID:
a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
```

Note: A *passphrase* is a sequence of words or other text, similar to a password, but generally longer for added security.

a. Save the Master Key to a file. Note that **-file** is a parameter and not the actual file name:

```
DCX95:admin>cryptocfg --exportmasterkey -file
Master key file generated.
```

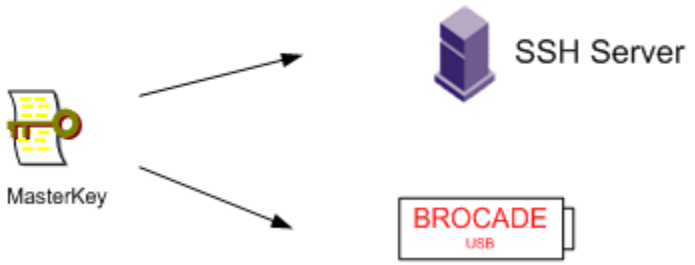


Figure 31 Master Key can be exported to different types of media

b. Export the Master Key to an SCP-capable external host or save it to the key vault:

```
DCX95:admin>cryptocfg --export -scp -currentMK 172.23.199.75 admin
/home/admin/certs/GL_MK.mk
admin1@172.23.199.75's password:
Operation succeeded
```

22. Verify the Group Leader is connected to the Key vault.

    Display the EG configuration to verify that the Group Leader is connected to the Key vault.

```
DCX95:admin>cryptocfg --show -groupcfg
Encryption Group Name:  brocade
    Failback mode:      Auto
    Heartbeat misses:   3
    Heartbeat timeout:  2
    Key Vault Type:     RKM

Primary Key Vault:
    IP address:         172.23.199.75
    Certificate ID:     RKMBrocade.Internal.com
    Certificate label:  RSA_cert
    State:              Connected
    Type:               RKM

Secondary Key Vault not configured

NODE LIST
Total Number of defined nodes:  1
Group Leader Node Name:         10:00:00:05:1e:46:08:00
Encryption Group state:         CLUSTER_STATE_CONVERGED

      Node Name                  IP address        Role
10:00:00:05:1e:46:08:00         172.23.199.22     GroupLeader  (current node)
```

23. Configure I/O synchronization links;

    Configure I/O synchronization links (Eth0 port) for private LAN communication between EEs.

    In this example, establish physical connections for eth0 and eth1 to private LAN:

```
DCX95:admin> ipaddrset -slot 4 -eth0 --add 172.23.101.10/24
DCX95:admin> ipaddrset -slot 4 -gate --add 172.23.101.1

DCX95:admin> ipaddrset -slot 12 -eth0 --add 172.23.101.11/24
DCX95:admin> ipaddrset -slot 12 -gate --add 172.23.101.1
```

    Note: The Eth0 and Eth1 ports are bonded together as a single virtual network interface that provides link layer redundancy. Only Ge0 needs to be configured.

24. View the IP address configuration.

```
DCX95:admin> ipaddrshow

CHASSIS
Ethernet IP Address: 172.23.199.22
Ethernet Subnetmask: 255.255.255.0

CP0
Ethernet IP Address: 172.23.199.23
Ethernet Subnetmask: 255.255.255.0
Host Name: cp0
Gateway IP Address: 172.23.199.2

CP1
Ethernet IP Address: 172.23.199.24
Ethernet Subnetmask: 255.255.255.0
Host Name: cp1
Gateway IP Address: 172.23.199.1

Slot 4
eth0: 172.23.101.10/24

Slot 12
eth0: 172.23.101.11/24

Backplane IP address of CP0 : 10.0.0.5
Backplane IP address of CP1 : 10.0.0.6
IPv6 Autoconfiguration Enabled: Yes
Local IPv6 Addresses:
IPv6 Gateways:
```

25. Enable EEs:

```
DCX95:admin>cryptocfg -enableEE 4
Operation succeeded.

DCX95:admin>cryptocfg -enableEE 12
Operation succeeded.
```

26. Verify that the EEs are enabled.

The SP state on both EEs should be online:

```
DCX95:admin>cryptocfg --show -groupmember -all

NODE LIST
Total Number of defined nodes:  1
Group Leader Node Name:         10:00:00:05:1e:46:08:00
Encryption Group state:         CLUSTER_STATE_CONVERGED

Node Name:                      10:00:00:05:1e:46:08:00  (current node)
State:                      DEF_NODE_STATE_DISCOVERED
Role:                       GroupLeader
    IP Address:                 172.23.199.22
    Certificate:                172.23.199.22_my_cp_cert.pem
    Current Master Key State:   Saved
    Current Master KeyID:       a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
    Alternate Master Key State: Not configured
    Alternate Master KeyID:     00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
   EE Slot:                   4
        SP state:               Online
        Current Master KeyID:   a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
        Alternate Master KeyID: 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
        HA Cluster Membership:
            Media Type      :   MEDIA NOT DEFINED
EE Slot:                    12
        SP state:               Online
        Current Master KeyID:   a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
        Alternate Master KeyID: 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
        HA Cluster Membership:
            Media Type      :   MEDIA NOT DEFINED
```

27. Verify connectivity of group leader node to key vault.

The state should be connected:

```
DCX95:admin> cryptocfg --show -groupcfg
Encryption Group Name:  brocade
    Failback mode:      Auto
    Heartbeat misses:   3
    Heartbeat timeout:  2
    Key Vault Type:     RKM

Primary Key Vault:
```

```
    IP address:          172.23.199.22
    Certificate ID:      RKMBrocade.Internal.com
    Certificate label:   RSA_cert
    State:               Connected
    Type:                RKM

Secondary Key Vault not configured

NODE LIST
Total Number of defined nodes:  1
Group Leader Node Name:         10:00:00:05:1e:46:08:00
Encryption Group state:         CLUSTER_STATE_CONVERGED

      Node Name                  IP address        Role
10:00:00:05:1e:46:08:00         10.66.19.95     GroupLeader  (current node)
```

Step 2 is now complete and the following checkpoint reached.

CHECKPOINT:

CX95 in Fabric A is configured with the RKM key vault and established as the group leader node.

Step 3: Initialize Encryption Node DCX98 in Fabric B

To initialize encryption node DCX98 in Fabric B, complete Step 1 through Step 11:

Detailed steps

1. Initialize encryption node DCX98 in Fabric B.

   This procedure must be completed once for all Encryption Group nodes being added to the EG; in this example, only DCX98 is initialized:

```
DCX98:admin>cryptocfg --initnode
This will overwrite all identification and authentication data
ARE YOU SURE (yes, y, no, n): [no] y
Notify SPM of Node Cfg
Operation succeeded.
```

2. Configure I/O synchronization links (Eth0 port) for private LAN communication between EEs.

   Establish physical connections for eth0 and eth1 to the private LAN:

```
DCX98:admin> ipaddrset -slot 4 -eth0 --add 172.23.101.12/24
DCX98:admin> ipaddrset -slot 4 -gate --add 172.23.101.1

DCX98:admin> ipaddrset -slot 12 -eth0 --add 172.23.101.13/24
DCX98:admin> ipaddrset -slot 12 -gate --add 172.23.101.1
```

Note: The Eth0 and Eth1ports are bonded together as a single virtual network interface that provides link layer redundancy. Only Ge0 needs to be configured.

3. Export the CP certificate to group leader node DCX95 in Fabric A.

   The following example is executed from DCX98 in Fabric B and exports the switch CP certificate from a member node to the SCP-capable host:

```
DCX98:admin>cryptocfg --export -scp -CPcert 172.23.199.75 admin
/home/admin/certs/DCX98_cp_cert.pem
admin1@172.23.199.75's password:
Operation succeeded.
```

4. From the GL node DCX95, import the member node's CP certificate file that was exported to the SCP-capable host in Step 3.

```
DCX95:admin> cryptocfg --import -scp DCX98_cp_cert.pem 172.23.199.75 admin
   /home/admin/certs/DCX98_cp_cert.pem
    Available Space:16384
    Make sure your file size is not greater than 20480.
   The switch will be unstable or the operation will fail if you exceed this limit.
    Do you want to proceed?
    ARE YOU SURE  (yes, y, no, n): [no] y
admin1@172.23.199.75's password:
Operation succeeded.
```

5. Verify the existing certificates.

   To verify existing certificates on either ED-DCX-B in Fabric A or B use the following. At this stage, the group leader node should have both the RKM and member node CP certificates:

```
DCX95:admin> cryptocfg --show -file -all

File name: my_cert.pem, size: 1338 bytes
File name: RKM_cert.pem,  size: 891 bytes       (RKM_cert.pem)
File name: DCX98_cp_cert.pem,    size: 1338 bytes (Member node CP cert)
```

6. Register the DCX98 to form the DEK cluster.

   From the group leader DCX95, register the DCX98 in Fabric B as a member node to the existing "brocade" EG, forming the DEK cluster.

   The **cryptoCfg** command registers the member node using the previously imported CP certificate.

   The command syntax is as follows:

```
DCX95:admin> cryptocfg --reg -membernode <member node WWN> <member node certfile>
   <IP address>
```

For example:

```
DCX95:admin> cryptocfg --reg -membernode 10:00:00:05:1e:46:50:00
   DCX98_cp_cert.pem 172.23.199.75
Operation succeeded.
```

7. Verify that the member node has successfully joined the EG.

```
DCX95:admin> cryptocfg --show -groupcfg
Encryption Group Name:  brocade
    Failback mode:      Auto
    Heartbeat misses:   3
    Heartbeat timeout:  2
    Key Vault Type:     RKM

Primary Key Vault:
    IP address:         172.23.199.22
    Certificate ID:     RKMBrocade.Internal.com
    Certificate label:  RSA_cert
    State:              Connected
    Type:               RKM

Secondary Key Vault not configured

NODE LIST
Total Number of defined nodes:  2
Group Leader Node Name:         10:00:00:05:1e:46:08:00
Encryption Group state:         CLUSTER_STATE_CONVERGED

      Node Name                  IP address        Role
10:00:00:05:1e:46:08:00         172.23.199.22     GroupLeader  (current node)
10:00:00:05:1e:46:50:00         172.23.199.75     MemberNode
```

8. Enable the EEs.

   The following commands initialize, register, enable, and verify that each of the EEs on the member node is enabled:

   - For the EE in slot 4:

```
DCX98:admin> cryptocfg --initEE 4
This will overwrite previously generated identification and authentication data
ARE YOU SURE  (yes, y, no, n): [no] y
Operation succeeded.

DCX98:admin> cryptocfg --regEE 4
Operation succeeded.

DCX98:admin> cryptocfg --enableEE 4
Operation succeeded
```

- For the EE in slot 12:

```
DCX98:admin> cryptocfg --initEE 12
This will overwrite previously generated identification and authentication data
ARE YOU SURE  (yes, y, no, n): [no] y
Operation succeeded.

DCX98:admin> cryptocfg --regEE 12
Operation succeeded.

DCX98:admin> cryptocfg --enableEE 12
Operation succeeded
```

9. Request the DCX98 switch certificate signing, import the signed certificate, and register the certificate on the member node

   a. Export the member node KAC certificate signing request to the link member node to the RKM key vault.

```
DCX98:admin>cryptocfg --export -scp -KACcsr 172.23.199.75 admin
   /home/admin/certs/DCX98_kac_cert.pem
### Get FN </etc/fabos/certs/sw0/kac_req.csr> ###
admin1@172.23.199.75's password:
Operation succeeded.
```

   b. As we did for the Encryption Group leader Node, we must now submit the member node KAC CSR (DCX98_kac_cert.pem) to a certificate signing authority for signing.

      For the purposes of this example, assume the signed DCX98 Member Node KAC certificate is called DCX98_kac_cert_signed.pem and stored on the SCP-capable host in the /home/admin/certs directory.

   c. Import the signed certificate file from the SCP-capable host:

```
DCX98:admin>cryptocfg --import -scp DCX98_kac_cert_signed.pem
   172.23.199.75 admin /home/admin/certs/DCX98_kac_cert_signed.pem
Available Space:16384
Make sure your file size is not greater than 16384.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to proceed?
ARE YOU SURE  (yes, y, no, n): [no] yes
admin1@172.23.199.75's password:
Operation succeeded.
```

   d. Register the signed imported KAC certificate:

```
DCX98:admin>cryptocfg --reg -KACcert DCX98_kac_cert_signed.pem
kac.00000000 051e465000
Register KAC status: Operation Succeeded.
```

10. In the RKM management GUI, create a new identify for the member node ED-DCX-B Fabric B switch.

    a. Click the **Identities** tab.

    b. Click **Create**.

    c. Enter the encryption blade switch name (DCX98, or any other name to help uniquely identify that Node/switch) into the **Name** field.

    d. Select **Hardware Retail Group** as the Identity Group.

    e. Select **Operational User** as the Role.

    f. Click **Browse** and select the signed "DCX98_kac_cert_signed.pem" as the Identity Certificate (the same signed "DCX98_kac_cert_signed.pem" from Step 9).

    g. Click **Save**.

    Note: The EG leader DCX95 in Fabric A has imported the CA certificate in "Step 2: Initialize Encryption Node DCX95 in Fabric A" on page 172. The CA certificate does not need to be imported into member nodes such as DCX98, as the EG leader will push it out to all member nodes in the EG.

11. Verify connectivity from the member node to the key vault.

    The NODE LIST should have two nodes, and the member node along with the group leader shows all SP states as online:

```
DCX98:admin> cryptocfg --show -groupmember -all

NODE LIST
Total Number of defined nodes:  2
Group Leader Node Name:         10:00:00:05:1e:46:08:00
Encryption Group state:         CLUSTER_STATE_CONVERGED


Node Name:                      10:00:00:05:1e:46:08:00
    State:                      DEF_NODE_STATE_DISCOVERED
    Role:                       GroupLeader
    IP Address:                 172.23.199.22
    Certificate:                172.23.199.75_my_cp_cert.pem
    Current Master Key State:   Saved
    Current Master KeyID:       a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
    Alternate Master Key State: Not configured
    Alternate Master KeyID:     00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

    EE Slot:                    4
```

```
        SP state:                 Online
        Current Master KeyID:     a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
        Alternate Master KeyID:   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
        HA Cluster Membership:
            Media Type      :     MEDIA NOT DEFINED
    EE Slot:                      12
        SP state:                 Online
        Current Master KeyID:     a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
        Alternate Master KeyID:   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
        HA Cluster Membership:
            Media Type      :     MEDIA NOT DEFINED

Node Name:                        10:00:00:05:1e:46:50:00  (current node)
    State:                        DEF_NODE_STATE_DISCOVERED
    Role:                         MemberNode
    IP Address:                   172.23.199.75
    Certificate:                  172.23.199.75_my_cp_cert.pem
    Current Master Key State:     Saved
    Current Master KeyID:         a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
    Alternate Master Key State:   Not configured
    Alternate Master KeyID:       00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

    EE Slot:                      4
        SP state:                 Online
        Current Master KeyID:     a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
        Alternate Master KeyID:   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
        HA Cluster Membership:
            Media Type      :     MEDIA NOT DEFINED

    EE Slot:                      12
        SP state:                 Online
        Current Master KeyID:     a7:45:63:48:0a:76:97:27:dd:30:67:cd:fb:68:c9:8f
        Alternate Master KeyID:   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
        No HA cluster membership
            Media Type      :     MEDIA NOT DEFINED
```

Step 3 is now complete and the following checkpoint reached.

CHECKPOINT:

DCX98 in Fabric B is configured with the RKM key vault and established as the member node.

Step 4: Configure HA clusters for EEs on DCX95 and DCX98 in Fabrics A and B

To configure HA clusters for EEs on DCX95 and DCX98 in Fabrics A and B, complete Step 1 through Step 3:

The following steps create two HA cluster pairs out of the two PB-DCX-16EB Encryption Blades in each ED-DCX-B. The HA clusters allow CTC failover between each encryption blade and are created

for full redundancy. Also note that when creating the HA cluster, use the same HA cluster name for the cluster pair.

The configuration below creates an HA cluster named "DCX95_CLUSTER" from the encryption blades in slots 4 and 12.

1. Configure an HA cluster for EEs in Fabric A DCX95:

   The command syntax is as follows:

```
DCX95:admin> cryptocfg --create -hacluster <HA cluster name> [<node WWN> [slot
   number]]+:
```

   For example:

```
DCX95:admin> cryptocfg --create -hacluster DCX95_CLUSTER 10:00:00:05:1e:46:08:00 4
                              Slot     Local/
      EE Node WWN           Number    Remote
10:00:00:05:1e:46:08:00       4       Local
Operation succeeded.
```

   Add failover encryption to the HA cluster and commit changes:

```
DCX95:admin> cryptocfg --add -haclustermember DCX95_CLUSTER 10:00:00:05:1e:46:08:00
   12
                               Slot     Local/
      EE Node WWN            Number    Remote
10:00:00:05:1e:46:08:00       12       Local
Operation succeeded.

DCX95:admin> cryptocfg --commit
Operation succeeded.
```

2. Verify the cluster creation, membership, and state.

```
DCX95:admin> cryptocfg --show -hacluster DCX95_CLUSTER
Encryption Group Name: brocade

HA cluster name: DCX95_CLUSTER - 2 EE entries
Status:          Committed
          WWN              Slot Number   Status
10:00:00:05:1e:46:08:00         4         Online
10:00:00:05:1e:46:08:00        12         Online
```

3. Repeat the HA cluster configuration steps for the DCX98 group member node in Fabric B called "DCX98_CLUSTER."

Step 4 is now complete and the following checkpoint reached.

CHECKPOINT:

There are now a total of two cluster pairs, DCX95_CLUSTER for encryption blades 4 and 12 in Fabric A and DCX98_CLUSTER for encryption blades 4 and 12 in Fabric B.

Step 5: Create CTCs in both Fabrics A and B

To create CTCs in both Fabrics A and B, complete Step 1 through Step 4:

Also included in this step are two optional configurations, along with steps to create them:

- "Optional configuration 1: Add a second initiator to the existing CTCs " on page 209.
- "Optional configuration 2: Create new CTCs in both Fabric A and B using Blue Storage 1 and 2 (LUN 0x26) for existing initiators Red Hosts 1 and 2 in Fabrics A and B using the other encryption blade in slot 4. Then add LUN 0x26 to both CTCs. " on page 214.

1. Create Crypto Target Containers (CTCs).

   Identify initiator and target ports with associated LUNs to be placed in CTCs for encryption in both Fabrics A and B for multipath access to LUNs.

   The first CTC in Fabric A DCX95 uses the encryption blade in slot 12, while the first CTC in Fabric B DCX98 also uses the encryption blade in slot 12, as shown in Figure 26 on page 165.

   Note: The following assumes that zoning is in place for initiator and storage ports.

   You need the following information to create a CTC:

   - Target World Wide Port Name (WWPN) and World Wide Node Name (WWNN) for all storage array Target ports through which LUNs will be encrypted
   - Initiator WWPN and WWNN
   - Encryption node WWN and slot number

   The following example uses the WWPN/WWNN from Red Storage 1 with Red Host HBA 1 for the CTC in Fabric A and the WWPN/WWNN of Red Storage 2 with Red Host HBA 2 for the CTC in Fabric B. Three LUNs (0x27,0x28 and 0x29) are added to both CTCs. The fourth LUN (0x31) will be configured for the Blue Host in later steps. Refer to the reference architecture in Figure 26 on page 165.

2. Obtain WWNs for devices to be used in the CTC for the Fabric A DCX95.

To obtain the WWNs needed for creating a CTC, use the following commands: **cfgShow**, **nodeFind**, or **wwn**.

- Use **cfgShow** to obtain WWNs for devices to be used in the CTC for Fabric A DCX95:

```
Effective configuration:
 cfg:   cfg
 zone:  Red_FAB_A
                10:00:00:05:1e:0c:1e:1b
                50:06:04:8a:d5:f0:c5:ae
(Output truncated)
```

- Use the **nodeFind** command to obtain both the WWPN and WWNN of both devices when both the target WWN and initiator WWNs are known:

```
DCX95:admin> nodefind 10:00:00:05:1e:0c:1e:1b
Remote:
    Type Pid    COS     PortName                NodeName
    N    5d0800;      3;10:00:00:05:1e:0c:1e:1b;20:00:00:05:1e:0c:1e:1b;
        FC4s: FCP
        PortSymb: [45] "Brocade-825 | 1.0.0.02. | E06-HP104-DCX |  | "
        Fabric Port Name: 20:08:00:05:1e:0a:15:49
        Permanent Port Name: 10:00:00:05:1e:0c:1e:1b
        Device type: Physical Initiator
        Port Index: 8
        Share Area: No
        Device Shared in Other AD: No
        Redirect: No
    Aliases: RedHost_BRCD_HBA1

DCX95:admin> nodefind 50:06:04:8a:d5:f0:c5:ae
Local:
 Type Pid    COS     PortName                NodeName                 SCR
 N    5f0400;      3;50:06:04:8a:d5:f0:c5:ae;50:06:04:8a:d5:f0:c5:ae; 3
    FC4s: FCP
    PortSymb: [38] "EMC SYMMETRIX 000190300950 SAF-15cB   "
    NodeSymb: [38] "EMC SYMMETRIX 000190300950 SAF-15cB   "
    Fabric Port Name: 20:04:00:05:1e:46:08:00
    Permanent Port Name: 50:06:04:8a:d5:f0:c5:ae
    Device type: Physical Initiator+Target
    Port Index: 4
    Share Area: No
    Device Shared in Other AD: No
    Redirect: No
    Aliases: SYM_A
```

- Use the **wwn** command to obtain the encryption node WWN for Fabric A DCX95:

```
DCX95:admin> wwn
10:00:00:05:1e:46:08:00
```

3. For Fabric A, on DCX95 (the Group Leader node), create a CTC for Red Storage 1 and add the RED Host HBA 1 initiator the CTC.

Note: For encryption solutions based on FOS 6.2.0x, the supported maximum number of CTCs per EE and/or HA cluster is one for access to any particular LUN. That is, a given LUN cannot be accessed within an EE or HA cluster by more than one CTC/target port.

Use the **cryptoCfg** command to create the CTC using the WWPN/WWNN for the devices and WWN of the encryption node with the slot number of the EE.

The command syntax is as follows:

```
DCX95:admin> cryptocfg --create –container <disk | tape> <crypto target container
   name>
<<node WWN> [slot number]> <Target WWPN> <Target WWNN>
```

For example:

```
DCX95:admin> cryptocfg --create -container disk SID_0950_15cB
   10:00:00:05:1e:46:08:00 12 50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
Operation succeeded.
```

4. Add the initiator to the target container.

   The command syntax is as follows:

```
DCX95:admin> cryptocfg --add –initiator <crypto target container name> <<Initiator
   WWPN> <Initiator WWNN>>
```

For example:

```
DCX95:admin> cryptocfg --add -initiator SID_0950_15cB 10:00:00:05:1e:0c:1e:1b
   20:00:00:05:1e:0c:1e:1b
Operation succeeded.
```

Note: The initiator is added at the CTC level to allow discovery of the LUNs behind the storage port to which the added initiator has access.

CHECKPOINT 5a:

The CTC is configured for Red Storage 1 and the Red Host HBA1 initiator is added to it, however the change has not yet taken effect. The **commit** command will be used in the next step to implement the configuration change.

Note: A maximum of 25 transactions per commit (**cryptocfg --commit**) can be executed.

To add a LUN, have the LUN numbers ready to use in the following steps.

Adding LUNs to a CTC is important since this is how the LUN can be accessed after the redirection zones are created, as shown in Figure 23 on page 156. There are two ways to add LUNs to a CTC:

- If the LUN numbers are known, this procedure can be non-disruptive as long as the host operating system and device drivers can accept Port Identifier (PID) changes as a result of frame redirection.

  If this is the case, go to “"Step 1a: Known LUN numbers" on page 195 and "Step 2a: Commit the CTC creation" on page 196.

OR

- If the LUN numbers are unknown, this procedure is disruptive to the I/O path. Since LUN numbers are needed, committing the CTC creation is required prior to LUN discovery, which exposes the LUNs on the target port.

  If this is the case, go to "Step 1b: Unknown LUN numbers" on page 196, and "Step 2b, Add desired LUNs to the CTC and commit the change" on page 197.

Step 1a: Known LUN numbers

If LUN numbers are known, add the LUNs to the CTC.

In this example, it is already known that LUNs 0x27, 0x28, 0x29 and 0x31 are exposed through the target port on which the CTC was created. The command uses the **LUN Num Range** option with the **cryptoCfg** command, since the first three LUNs are in sequential order.

The following example adds only three of the four LUNs. The last LUN 0x31 is reserved for a subsequent configuration.

The command syntax is as follows:

```
DCX95:admin> cryptocfg --add -LUN <crypto target container name> <LUN Num | LUN
   Num Range> <<Initiator WWPN> <Initiator WWNN>>
```

For example:

```
DCX95:admin> cryptocfg --add -LUN SID_0950_15cB 0x27-0x29 10:00:00:05:1e:0c:1e:1b
   20:00:00:05:1e:0c:1e:1b
Operation succeeded.
```

Step 2a: Commit the CTC creation

```
DCX95:admin> cryptocfg --commit
Operation succeeded.
```

Use **cryptocfg –commit** to commit all previous transactions.

CHECKPOINT 5b:

The CTC named SID_0950_15cB is configured from target port 50:06:04:8a:d5:f0:c5:ae with initiator 10:00:00:05:1e:0c:1e:1b and LUNs 0x27, 0x28 and 0x29. The initiator still has access to the LUNs. The host may have experienced a slight I/O disruption in the PID change during Frame Redirection; however it still has continuous access to the LUNs.

Skip "Step 1b: Unknown LUN numbers" on page 196, and "Step 2b, Add desired LUNs to the CTC and commit the change" on page 197 if "Step 1a: Known LUN numbers" on page 195 and "Step 2a: Commit the CTC creation" on page 196 were performed.

Step 1b: Unknown LUN numbers

If LUN numbers are unknown, commit the transaction after adding the initiator for LUN discovery:

```
DCX95:admin> cryptocfg --commit
Operation succeeded.
```

CHECKPOINT 5c:

The CTC named SID_0950_15cB has been created from target port 50:06:04:8a:d5:f0:c5:ae with initiator 10:00:00:05:1e:0c:1e:1b without adding any LUNs. This means that the initiator no longer has access to the LUNs on the target port.

Once the CTC has been created and the configuration committed, use the **cryptocfg –discoverLUN** command for LUN discovery.

The command syntax is as follows:

```
DCX95:admin> cryptocfg --discoverLUN <crypto target container name>
```

For example:

```
DCX95:admin> cryptocfg --discoverLUN SID_0950_15cB
Container name:          SID_0950_15cB
Number of LUN(s):                5
Host:                    10:00:00:05:1e:0c:1e:1b
LUN number:              0x0
LUN serial number:       60060480000190300950533030303230
Key ID state:            Key ID not available

Host:                    10:00:00:05:1e:0c:1e:1b
```

```
LUN number:              0x27
LUN serial number:       600604800001903009505330303045 35
Key ID state:            Key ID not Applicable

Host:                    10:00:00:05:1e:0c:1e:1b
LUN number:              0x28
LUN serial number:       600604800001903009505330303045 36
Key ID state:            Key ID not Applicable

Host:                    10:00:00:05:1e:0c:1e:1b
LUN number:              0x29
LUN serial number:       600604800001903009505330303131 34
Key ID state:            Key ID not Applicable

Host:                    10:00:00:05:1e:0c:1e:1b
LUN number:              0x31
LUN serial number:       600604800001903009505330303040 87
Key ID state:            Key ID not Applicable

Operation succeeded.
```

Step 2b, Add desired LUNs to the CTC and commit the change

Once the LUNs are discovered with their numbers, add the desired LUNs to the CTC and commit the change. The following example adds LUNs 0x27, 0x28 and 0x29 using the **range** option since they are in sequential order.

```
DCX95:admin> cryptocfg --add -LUN SID_0950_15cB 0x27-0x29 10:00:00:05:1e:0c:1e:1b
   20:00:00:05:1e:0c:1e:1b
Operation succeeded.
```

Note: The following is an example and *does not actually add LUN 0x31 to the configuration at this time.* The example shows how to add a single LUN. Using the range option does not work on LUN 0x31 since it is not part of the sequence of numbers (0x27-0x29) specified in the range from the previous command.

```
DCX95:admin> cryptocfg --add -LUN SID_0950_15cB 0x31 10:00:00:05:1e:0c:1e:1b
   20:00:00:05:1e:0c:1e:1b
Operation succeeded.
```

Commit the transactions:

```
DCX95:admin> cryptocfg --commit
Operation succeeded
```

CHECKPOINT 5d:

The LUNs are added to the existing CTC SID_0950_15cB. The initiator once again has access to the LUNs. For operating systems that keep track of the detailed path through the fabric to a LUN, such

as AIX (cfgmgr) or HP-UX (ioscan), it may be necessary to perform a LUN discovery to make them visible to the initiator.

In summary, use "Step 1a: Known LUN numbers" on page 195 and "Step 2a: Commit the CTC creation" on page 196 to add LUNs non-disruptively when LUN numbers are known, if the host operating system and device drivers can take the PID change when redirection zones are created. Alternatively, use "Step 1b: Unknown LUN numbers" on page 196, and "Step 2b, Add desired LUNs to the CTC and commit the change" on page 197 when LUN numbers are unknown.

Note: When adding LUNs to a CTC using either procedure, if no options are used in the **cryptoCfg –add –LUN** command, the default action is to add the LUNs to the CTC as cleartext.

This is the recommended action and best practice when configuring a LUN and preparing it for first-time encryption.

Verify that the LUNs were successfully added to the CTC, as shown next:.

```
DCX95:admin> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 1

Container name:          SID_0950_15cB
Type:                    disk
EE node:                 10:00:00:05:1e:46:08:00
EE slot:                 12
EE hosting container:    current
Target:                  50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
Target PID:              5f0400
VT:                      20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
VT PID:                  5ff601
Number of host(s):       1
Number of rekey session(s):     0
Host:                    10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
Host PID:                5d0800
VI:                      20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
VI PID:                  5ff801
Number of LUN(s):           3   (Verify the number of LUNs)

LUN number:              0x27
LUN type:                disk
LUN serial number:       60060480000190300950533030304535
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
```

```
Rekey:                    disabled
Internal EE LUN state:    Clear text
Encryption algorithm:     None
Key ID state:             Key ID not Applicable

LUN number:               0x28
LUN type:                 disk
LUN serial number:        60060480000190300950533030304536
Encryption mode:          cleartext
Encryption format:        native
Encrypt existing data:    disabled
Rekey:                    disabled
Internal EE LUN state:    Clear text
Encryption algorithm:     None
Key ID state:             Key ID not Applicable

LUN number:               0x29
LUN type:                 disk
LUN serial number:        60060480000190300950533030313134
Encryption mode:          cleartext
Encryption format:        native
Encrypt existing data:    disabled
Rekey:                    disabled
Internal EE LUN state:    Clear text
Encryption algorithm:     None
Key ID state:             Key ID not Applicable
Operation succeeded.
```

CHECKPOINT 5e:

All three LUNs are successfully added to CTC SID_0950_15cB as cleartext. At this point all data between the initiator and target LUNs defined are now passing through the EE in cleartext.

1. Obtain WWNs for devices to be used in the CTC for the Fabric B DCX98.

   The following WWNs are collected for Red Storage 2 and Red Host HBA 2 for the member node DCX98 in Fabric B as shown in Figure 26 on page 165.

   The following is truncated output from **cfgshow** for Fabric B DCX98:

```
DCX98:root> cfgshow

Effective configuration:
cfg:   cfg
zone:  Red_FAB_B
10:00:00:05:1e:0c:1e:1c
50:06:04:8a:d5:f0:c5:a1
```

When both the target WWN and initiator WWNs are known, use the **nodeFind** command to obtain both the WWPN and WWNN of both devices:

```
DCX98:root> nodefind 10:00:00:05:1e:0c:1e:1c
Remote:
    Type Pid    COS     PortName                NodeName
    N    5b0800;      3;10:00:00:05:1e:0c:1e:1c;20:00:00:05:1e:0c:1e:1c;
        FC4s: FCP
        PortSymb: [45] "Brocade-825 | 1.0.0.02. | E06-HP104-DCX |  | "
        Fabric Port Name: 20:08:00:05:1e:58:11:8c
        Permanent Port Name: 10:00:00:05:1e:0c:1e:1c
        Device type: Physical Initiator
        Port Index: 8
        Share Area: No
        Device Shared in Other AD: No
        Redirect: Yes host
    Aliases: RedHost_BRCD_HBA2

DCX98:root> nodefind 50:06:04:8a:d5:f0:c5:a1
Local:
 Type Pid    COS     PortName                NodeName                 SCR
 N    620400;      3;50:06:04:8a:d5:f0:c5:a1;50:06:04:8a:d5:f0:c5:a1; 3
    FC4s: FCP
    PortSymb: [38] "EMC SYMMETRIX 000190300950 SAF- 2cB   "
    NodeSymb: [38] "EMC SYMMETRIX 000190300950 SAF- 2cB   "
    Fabric Port Name: 20:04:00:05:1e:46:50:00
    Permanent Port Name: 50:06:04:8a:d5:f0:c5:a1
    Device type: Physical Initiator+Target
    Port Index: 4
    Share Area: No
    Device Shared in Other AD: No
    Redirect: Yes target
    Aliases: RedStor_EMC_FABB
```

2. Create a CTC from a specified target port; add an initiator and LUNs for devices in Fabric B, DCX98.

Note: All **cryptoCfg** commands must be performed from the Group Leader Node DCX95 in Fabric A even if the CTC being created is on Group Member Node DCX98 in Fabric B.

For example, if the **cryptoCfg** command is executed from the member node, the following message is displayed:

"Operation failed: Crypto device configuration change is not allowed at non-GL node"

The **cryptoCfg** command can be executed from member nodes *only if it is displaying or querying the existing configuration*.

The following series of commands create the container, add the LUNs, and commit to the CTC for the Member Node DCX98 in Fabric B:

```
DCX95:admin> cryptocfg --create -container disk SID_0950_2cB
   10:00:00:05:1e:46:50:00 12 50:06:04:8a:d5:f0:c5:a1 50:06:04:8a:d5:f0:c5:a1

DCX95:admin> cryptocfg --add -initiator SID_0950_2cB 10:00:00:05:1e:0c:1e:1c
   20:00:00:05:1e:0c:1e:1c

DCX95:admin> cryptocfg --add -LUN SID_0950_2cB 0x27-0x29 10:00:00:05:1e:0c:1e:1c
   20:00:00:05:1e:0c:1e:1c

DCX95:admin> cryptocfg -commit
Operation succeeded.
```

CHECKPOINT 5f:

The CTC for Fabric B contains the corresponding initiator, target and LUNs that were placed in the CTC for Fabric A to ensure multipath functionality.

1. Verify that the CTCs in Fabrics A and B are successfully created with their respective LUNs added.

   Once both CTCs (one in Fabric A DCX95 and one in Fabric B DCX98) have been configured with the same cleartext LUNs for both fabrics, verify that the same LSNs were added.

   a. From Fabric A DCX95 Group Leader:

```
DCX95:admin> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 1

Container name:          SID_0950_15cB
Type:                    disk
EE node:                 10:00:00:05:1e:46:08:00
EE slot:                 12
EE hosting container:    current
Target:                  50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
Target PID:              5f0400
VT:                      20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
VT PID:                  5ff601
Number of host(s):       1
Number of rekey session(s):     0
Host:                    10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
Host PID:                5d0800
VI:                      20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
VI PID:                  5ff801
Number of LUN(s):        3  (Verify the number of LUNs)
```

```
LUN number:              0x27
LUN type:                disk
LUN serial number:         60060480000190300950533030304535 (LSN)
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text                          (LUN state)
Encryption algorithm:    None
Key ID state:            Key ID not Applicable

LUN number:              0x28
LUN type:                disk
LUN serial number:         60060480000190300950533030304536 (LSN)
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text    (LUN state)
Encryption algorithm:    None
Key ID state:            Key ID not Applicable

LUN number:              0x29
LUN type:                disk
LUN serial number:         60060480000190300950533030313134 (LSN)
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text
Encryption algorithm:    None
Key ID state:            Key ID not Applicable
Operation succeeded.
```

b. From Fabric B DCX98 Group Member:

```
DCX98:root> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 1

Container name:          SID_0950_2cB
Type:                    disk
EE node:                 10:00:00:05:1e:46:50:00
EE slot:                 12
EE hosting container:    current
Target:                  50:06:04:8a:d5:f0:c5:a1 50:06:04:8a:d5:f0:c5:a1
Target PID:              620400
VT:                      20:03:00:05:1e:54:17:49 20:03:00:05:1e:54:17:49
VT PID:                  62f801
Number of host(s):       1
Number of rekey session(s):     0
Host:                    10:00:00:05:1e:0c:1e:1c 20:00:00:05:1e:0c:1e:1c
```

```
Host PID:                5b0800
VI:                      20:04:00:05:1e:54:17:49 20:05:00:05:1e:54:17:49
VI PID:                  62f601
Number of LUN(s):          3   (Verify the number of LUNs)

LUN number:              0x27
LUN type:                disk
LUN serial number:         600604800001903009505330303045 35 (LSN)
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text    (LUN state)
Encryption algorithm:    None
Key ID state:            Key ID not Applicable

LUN number:              0x28
LUN type:                disk
LUN serial number:         600604800001903009505330303045 36 (LSN)
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text                        (LUN state)
Encryption algorithm:    None
Key ID state:            Key ID not Applicable

LUN number:              0x29
LUN type:                disk
LUN serial number:         600604800001903009505330303131 34 (LSN)
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text          (LUN state)
Encryption algorithm:    None
Key ID state:            Key ID not Applicable
Operation succeeded.
```

Note: The LUN number, LUN serial number, and internal EE LUN state are important to verify to ensure that the same LSNs have been added to both CTCs across fabrics. Verify the LUN state of the LUNs in each CTC.

2. Modify LUNs to enable encryption and preserve existing data in both CTCs for Fabrics A and B simultaneously.

   All **cryptoCfg** commands must be performed from the Group Leader Node DCX95 in Fabric A even if the CTC being modified is on Group Member Node DCX98 in Fabric B.

Enabling encryption and preserving the existing data on the LUNs requires First Time Encryption (FTE). This step uses the **cryptoCfg –modify** on both CTCs. Keep in mind that **cryptocfg** commands are executed only on the group leader node in Fabric A. Depending on the size of the LUNs and I/O access patterns of the application accessing data, the FTE process may take several hours or more.

IMPORTANT

It is strongly recommended to minimize I/O to the LUNs to ensure for optimal FTE performance.

Preserving the existing data reads the existing cleartext data on the LUN and writes it back in an encrypted form. The **-enable_encexistingdata** option is necessary, since there is existing data on the LUN to be preserved. Alternatively, to enable a LUN for encryption without having to preserve existing data or FTE, use the **-disable_encexistingdata** option.

The command syntax is as follows:

```
DCX95:admin> cryptocfg --modify -LUN <crypto target container name> <LUN Num>
<Initiator WWPN>[-encryption_format <native | DF_compatible>][-encrypt |
   -cleartext]
[-enable_encexistingdata | -disable_encexistingdata][-enable_rekey <time period>
   | -disable_rekey]
```

The following example does not use all of the options for this command. The command modifies LUN 0x27 on the CTC SID_0950_15cB in Fabric A to encrypt the LUN and to preserve the existing data for the Red Host HBA 1 WWN 10:00:00:05:1e:0c:1e:1b in the CTC.

- From Group Leader DCX95:

```
DCX95:admin> cryptocfg --modify -LUN SID_0950_15cB 0x27 10:00:00:05:1e:0c:1e:1b
   -encrypt -enable_encexistingdata
Operation succeeded.
```

The following command also modifies LUN 0x27, but for CTC SID_0950_2cB in the Fabric B member node to encrypt the LUN and to preserve the existing data for the Red Host HBA 2 WWN 10:00:00:05:1e:0c:1e:1c in the CTC.

- Also from Group Leader DCX95:

```
DCX95:admin> cryptocfg --modify -LUN SID_0950_2cB 0x27 10:00:00:05:1e:0c:1e:1c
   -encrypt -enable_encexistingdata
Operation succeeded.
```

Note: To avoid potential data corruption, modify all paths with identical configurations for a given LUN prior to using the **cryptoCfg --commit** command.

The **cryptoCfg –commit** command is extremely important, since it is used for modification of both LUN 0x27 in CTC SID_0950_15cB in Fabric A and for CTC SID_0950_2cB in Fabric B.

```
DCX95:admin> cryptocfg -- commit
Operation succeeded.
```

Once committed, both CTCs in Fabrics A and B will be in FTE mode for the specified LUNs.

CHECKPOINT 5g:

The same set of three LUNs are added to the CTCs in both fabrics. The **cryptocfg –modify** command was used for the LUNs in both CTCs simultaneously to ensure continuity. Also note that the LUN numbers along with the LSNs are the same from both CTCs between fabrics.

To monitor first-time rekeying of modified LUNs, show the status of any rekeying operation (First Time Rekey (FTR) or subsequent rekeying) issue the **cryptocfg --show -rekey -all** command as shown next.

```
DCX95:admin> cryptocfg --show -rekey -all

Container name:          SID_0950_15cB
EE node:                 10:00:00:05:1e:46:08:00
EE slot:                 12
Target:                  50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
Target PID:              5f0400
VT:                      20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
VT PID:                  5ff401
Host:                    10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
Host PID:                5d0800
VI:                      20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
VI PID:                  5ff601
LUN number:                 0x27                                    (LUN #)
LUN serial number:       60060480000190300950533030304535             (LSN)
Rekey session number:    0
Percentage complete:        12                                (Rekey  Status)
Rekey state:             Cluster Sync After Write Phase
```

```
Rekey role:              Primary/Active                          (Rekey role)
Block size:              512
Number of blocks:        23568000
Current LBA:             2961137
Operation succeeded.
Number of rekey session(s):     1
```

From the Member Node DCX98 in Fabric B, the view is essentially the same except that the rekey is actually being performed by the group leader as shown in the Rekey Role: Primary/Active from the output above versus the "Rekey Role: Secondary/Standby" output, below:

```
DCX98:admin> cryptocfg --show -rekey -all

Container name:          SID_0950_ 2cB
EE node:                 10:00:00:05:1e:46:50:00
EE slot:                 12
Target:                  50:06:04:8a:d5:f0:c5:a1 50:06:04:8a:d5:f0:c5:a1
Target PID:              620400
VT:                      20:03:00:05:1e:54:17:49 20:03:00:05:1e:54:17:49
VT PID:                  62f801
Host:                    10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
Host PID:                5d0800
VI:                      20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
VI PID:                  5ff601
LUN number:              0x27                                      (LUN #)
LUN serial number:       600604800001903009505330303045 35          (LSN)
Rekey session number:    0
Percentage complete:     12                                  (Rekey status)
Rekey state:             Cluster Member: Write Phase Done
Rekey role:              Backup/Redundant                        (Rekey role)
Block size:              512
Number of blocks:        23568000
Current LBA:             2961137
Operation succeeded.
Number of rekey session(s):     1
```

To monitor the progress of the FTE or rekey, check the "Percentage complete" section of the output.

In summary, a DEK is generated by the EE or "crypto-module" upon request or essentially when a LUN is set to be encrypted. In this case, a DEK was generated when the "cryptocfg –modify" and "cryptocfg –commit" was executed for both CTCs containing LUN 0x27 in the previous steps above.

Metadata, also labeled as the "key id," is written to the LUN as the identifier for the DEK that was generated for LUN 0x27 then stored to the RKM key vault. The FTE or rekey for a specific LUN is performed *only* by the EE that generates the DEK for that LUN. The EE

performing the FTE or rekey can be identified by the "Rekey Role" as the "Primary/Active" as shown in the previous outputs. The other EE not performing the FTE or rekey with access to the same LUN is identified by the "Rekey Role" as the "Backup/Redundant."

CHECKPOINT 5h:

When a first-time rekeying is initiated, only one EE performs the rekeying process.

To confirm the successful FTE/rekeying of modified LUNs, complete the following steps:

1. From the Group Leader node in Fabric A, show the status of the container with modified LUNs after rekeying.

```
DCX95:admin> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 1

Container name:          SID_0950_15cB
Type:                    disk
EE node:                 10:00:00:05:1e:46:08:00
EE slot:                 12
EE hosting container:    current
Target:                  50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
Target PID:              5f0400
VT:                      20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
VT PID:                  5ff401
Number of host(s):       1
Number of rekey session(s):     0
Host:                    10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
Host PID:                5d0800
VI:                      20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
VI PID:                  5ff601
Number of LUN(s):        3

LUN number:              0x27                                    ←
LUN type:                disk
LUN serial number:       60060480000190300950533030304535        ←
Encryption mode:         encrypt
Encryption format:       native
Encrypt existing data:   enabled                                 ←
Rekey:                   disabled
Internal EE LUN state:   Encryption enabled                       ←
Encryption algorithm:    AES256-XTS
Key ID state:            Read write
Key ID:                  bc:c3:ef:b8:55:db:35:2c:5a:c9:ad:9d:c2:09:30:d8  ←
Key creation time:       Tue May 12 23:18:12 2009                         ←

   LUN number:              0x28
LUN type:                disk
```

```
LUN serial number:      600604800001903009505330303045 36
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable

LUN number:             0x29
LUN type:               disk
LUN serial number:      600604800001903009505330303131 34
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable
Operation succeeded.
```

2. From the Group Member Node DCX98 in Fabric B, show the status of the container with modified LUNs after rekeying:

```
DCX98:root> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 1

Container name:         SID_0950_2cB
Type:                   disk
EE node:                10:00:00:05:1e:46:50:00
EE slot:                12
EE hosting container:   current
Target:                 50:06:04:8a:d5:f0:c5:a1 50:06:04:8a:d5:f0:c5:a1
Target PID:             620400
VT:                     20:03:00:05:1e:54:17:49 20:03:00:05:1e:54:17:49
VT PID:                 62f801
Number of host(s):      1
Number of rekey session(s):     0
Host:                   10:00:00:05:1e:0c:1e:1c 20:00:00:05:1e:0c:1e:1c
Host PID:               5b0800
VI:                     20:04:00:05:1e:54:17:49 20:05:00:05:1e:54:17:49
VI PID:                 62f601
Number of LUN(s):       3

LUN number:             0x27                                          ←
LUN type:               disk
LUN serial number:      600604800001903009505330303045 35             ←
Encryption mode:        encrypt
Encryption format:      native
Encrypt existing data:  enabled                                       ←
Rekey:                  disabled
```

```
Internal EE LUN state:  Encryption enabled                          ←
Encryption algorithm:   AES256-XTS
Key ID state:           Read write
Key ID:                 bc:c3:ef:b8:55:db:35:2c:5a:c9:ad:9d:c2:09:30:d8  ←
Key creation time:      Tue May 12 23:18:12 2009                    ←


LUN number:             0x28
LUN type:               disk
LUN serial number:      60060480000190300950533030304536
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable

LUN number:             0x29
LUN type:               disk
LUN serial number:      60060480000190300950533030313134
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable
Operation succeeded.
```

The previous output shows one of three LUNs (0x27) encrypted with the other two LUNS (0x28) and (0x29) still in the cleartext state. If the remaining LUNs are to be used for encryption, **cryptoCfg –modify** must be used to enable encryption for each CTC in Fabric A and B as shown for LUN 0x27 in the previous steps.

CHECKPOINT 5i:

All of the LSNs for added LUNs 0x27, 0x28 and 0x29 are identical to both CTCs in Fabrics A and B. Once all of the LUNs to be encrypted have been successfully modified for both CTCs, the initial configuration is complete.

Optional configuration 1: Add a second initiator to the existing CTCs

The following instructions add a second initiator Blue Host to the existing CTCs in Fabrics A and B and defines a new LUN 0x31 to be accessed by the Blue Host. This example displays how initiators can be added to a CTC and new LUNs defined.

1. Add Blue Host HBA 1 to existing CTC SID_0950_15cB in Fabric A DCX95:

```
DCX95:admin> cryptocfg --add -initiator SID_0950_15cB 21:00:00:1b:32:01:59:1b
   20:00:00:1b:32:01:59:1b
Operation succeeded.

DCX95:admin> cryptocfg --commit
Operation succeeded.
```

2. Verify that Blue Host HBA 1 initiator 21:00:00:1b:32:01:59:1b has been added to the CTC:

```
DCX95:admin> cryptocfg --show -container SID_0950_15cB -cfg
Container name:          SID_0950_15cB
Type:                    disk
EE node:                 10:00:00:05:1e:46:08:00
EE slot:                 12
Target:                  50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
VT:                      20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
Number of host(s):       2 (Both hosts are now added to the CTC)
Configuration status:    committed
Host:                    10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
                                                              (Red Host 1)
VI:                      20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
Number of LUN(s):        3
Host:                    21:00:00:1b:32:01:59:1b 20:00:00:1b:32:01:59:1b
                                                               Blue Host 1)
VI:                      20:06:00:05:1e:54:17:49 20:07:00:05:1e:54:17:49
Number of LUN(s):        0
```

Note that there are currently no LUNs defined for the Blue Host 1 initiator.

3. Add LUN 0x31 to Blue Host 1 initiator:

```
DCX95:admin> cryptocfg --add -LUN SID_0950_15cB 0x31 21:00:00:1b:32:01:59:1b
   20:00:00:1b:32:01:59:1b
Operation succeeded.

Commit the transactions:
DCX95:admin> cryptocfg --commit
Operation succeeded
```

4. Verify that the LUN has been successfully added:

```
DCX95:admin> cryptocfg --show -container SID_0950_15cB -cfg
Container name:          SID_0950_15cB
Type:                    disk
EE node:                 10:00:00:05:1e:46:08:00
EE slot:                 12
Target:                  50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
VT:                      20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
```

```
Number of host(s):      2
Configuration status:   committed
Host:                   10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
VI:                     20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
Number of LUN(s):       3
Host:                   21:00:00:1b:32:01:59:1b 20:00:00:1b:32:01:59:1b
VI:                     20:06:00:05:1e:54:17:49 20:07:00:05:1e:54:17:49
Number of LUN(s):          1            (Added LUN)
```

5. Verify that LUN 0x31 has been added to CTC SID_0950_15cB defined for initiator 21:00:00:1b:32:01:59:1b in cleartext state:

```
DCX95:admin> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 1
Container name:         SID_0950_15cB
Type:                   disk
EE node:                10:00:00:05:1e:46:08:00
EE slot:                12
EE hosting container:   current
Target:                 50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae
Target PID:             5f0400
VT:                     20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
VT PID:                 5ff601
Number of host(s):      2
Number of rekey session(s):     0
Host:                   10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
Host PID:               5d0800
VI:                     20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
VI PID:                 5ff801
Number of LUN(s):       3

LUN number:             0x27
LUN type:               disk
LUN serial number:      600604800001903009505330303045 35
Encryption mode:        encrypt
Encryption format:      native
Encrypt existing data:  enabled
Rekey:                  disabled
Internal EE LUN state:  Encryption enabled
Encryption algorithm:   AES256-XTS
Key ID state:           Read write
Key ID:                 bc:c3:ef:b8:55:db:35:2c:5a:c9:ad:9d:c2:09:30:d9
Key creation time:      Mon Jun  8 23:03:21 2009

LUN number:             0x28
LUN type:               disk
LUN serial number:      600604800001903009505330303045 36
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
```

```
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable

LUN number:             0x29
LUN type:               disk
LUN serial number:      60060480000190300950533030313134
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable

Host:        21:00:00:1b:32:01:59:1b 20:00:00:1b:32:01:59:1b (Blue Host 1)
Host PID:               5d0900
VI:                     20:06:00:05:1e:54:17:49 20:07:00:05:1e:54:17:49
VI PID:                 5ff001
Number of LUN(s):       1

LUN number:               0x31       (LUN)
LUN type:               disk
LUN serial number:      60060480000190300950533030304087        (LSN)
Encryption mode:        cleartext                       (added as cleartext)
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable
   Operation succeeded.
```

6. Repeat the steps for Blue Host HBA 2 in Fabric B DCX98.

Note: All **cryptoCfg** execution commands must be performed from the Group Leader Node DCX95 in Fabric A, even if the CTC being modified is on Group Member Node DCX98 in Fabric B.

From Group Leader Node Fabric A, DCX95:

```
DCX95:admin> cryptocfg --add -initiator SID_0950_2cB 21:01:00:1b:32:21:59:1b
   20:01:00:1b:32:21:59:1b
Operation succeeded.

DCX95:admin>  cryptocfg --commit
Operation succeeded.
```

7. Add LUN 0x31 to the initiator:

```
DCX95:admin> cryptocfg --add -LUN SID_0950_2cB 0x31 21:01:00:1b:32:21:59:1b
   20:01:00:1b:32:21:59:1b
Operation succeeded.
```

8. Commit the transactions:

```
DCX95:admin> cryptocfg --commit
Operation succeeded
```

9. Verify that LUN 0x31 has been added to CTC SID_0950_2cB defined for initiator 21:01:00:1b:32:21:59:1b in cleartext state.

```
DCX98:root> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 1

Container name:          SID_0950_2cB
Type:                    disk
EE node:                 10:00:00:05:1e:46:50:00
EE slot:                 12
EE hosting container:    current
Target:                  50:06:04:8a:d5:f0:c5:a1 50:06:04:8a:d5:f0:c5:a1
Target PID:              620400
VT:                      20:03:00:05:1e:54:17:49 20:03:00:05:1e:54:17:49
VT PID:                  62f801
Number of host(s):       2
Number of rekey session(s):     0
Host:                    10:00:00:05:1e:0c:1e:1c 20:00:00:05:1e:0c:1e:1c
Host PID:                5b0800
VI:                      20:04:00:05:1e:54:17:49 20:05:00:05:1e:54:17:49
VI PID:                  62f601
Number of LUN(s):        3
LUN number:              0x27
LUN type:                disk
LUN serial number:       60060480000190300950533030304535
Encryption mode:         encrypt
Encryption format:       native
Encrypt existing data:   enabled
Rekey:                   disabled
Internal EE LUN state:   Encryption enabled
Encryption algorithm:    AES256-XTS
Key ID state:            Read write
Key ID:                  bc:c3:ef:b8:55:db:35:2c:5a:c9:ad:9d:c2:09:30:d9
Key creation time:       Mon Jun  8 23:03:21 2009
LUN number:              0x28
LUN type:                disk
LUN serial number:       60060480000190300950533030304536
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text
Encryption algorithm:    None
Key ID state:            Key ID not Applicable
```

```
LUN number:              0x29
LUN type:                disk
LUN serial number:       60060480000190300950533030313134
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text
Encryption algorithm:    None
Key ID state:            Key ID not Applicable

Host:      21:01:00:1b:32:21:59:1b 20:01:00:1b:32:21:59:1b (Blue Host 2)
Host PID:                5b0900
VI:                      20:08:00:05:1e:54:17:49 20:09:00:05:1e:54:17:49
VI PID:                  62fa01
Number of LUN(s):        1

LUN number:              0x31                                        (LUN)
LUN type:                disk
LUN serial number:    60060480000190300950533030304087(LSN)
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text          (Added as cleartext)
Encryption algorithm:    None
Key ID state:            Key ID not Applicable
Operation succeeded.
```

Note: To modify LUN 0x31 for FTE in both CTCs to enable encryption, follow the procedure in Step 2, "Modify LUNs to enable encryption and preserve existing data in both CTCs for Fabrics A and B simultaneously. " on page 203.

CHECKPOINT 5j:

LUN 0x31 has been added to the CTCs in both Fabric A and B and has been set up to be accessed by Blue Host HBA 1 and Blue Host HBA 2. This configuration is now complete.

Optional configuration 2: Create new CTCs in both Fabric A and B using Blue Storage 1 and 2 (LUN 0x26) for existing initiators Red Hosts 1 and 2 in Fabrics A and B using the other encryption blade in slot 4. Then add LUN 0x26 to both CTCs.

Note: This example shows that an existing initiator with access to a LUN through a given CTC can access other CTCs on other EEs to load balance data flows.

1. Use the **cfgShow** command to obtain the WWNs of devices to be used in new CTC for Fabric A DCX95:

```
DCX95:admin> cfgshow
   RedHost_BlueStorA
                10:00:00:05:1e:0c:1e:1b
                50:06:04:8a:d5:f0:c5:8f
(Output truncated)
```

2. When both the target WWN and initiator WWNs are known, obtain both the WWPN and WWNN of both devices as previously mentioned using the **nodeFind** command:

```
DCX95:admin> nodefind 10:00:00:05:1e:0c:1e:1b
Remote:
    Type Pid    COS     PortName                NodeName
    N    5d0800;      3;10:00:00:05:1e:0c:1e:1b;20:00:00:05:1e:0c:1e:1b;
        FC4s: FCP
        PortSymb: [45] "Brocade-825 | 1.0.0.02. | E06-HP104-DCX |  | "
        Fabric Port Name: 20:08:00:05:1e:0a:15:49
        Permanent Port Name: 10:00:00:05:1e:0c:1e:1b
        Device type: Physical Initiator
        Port Index: 8
        Share Area: No
        Device Shared in Other AD: No
        Redirect: Yes host
    Aliases: RedHost_BRCD_HBA1

DCX95:admin> nodefind 50:06:04:8a:d5:f0:c5:8f
Local:
 Type Pid    COS     PortName                NodeName                SCR
 N    5f0500;      3;50:06:04:8a:d5:f0:c5:8f;50:06:04:8a:d5:f0:c5:8f; 3
    FC4s: FCP
    PortSymb: [38] "EMC SYMMETRIX 000190300950 SAF-16cA   "
    NodeSymb: [38] "EMC SYMMETRIX 000190300950 SAF-16cA   "
    Fabric Port Name: 20:05:00:05:1e:46:08:00
    Permanent Port Name: 50:06:04:8a:d5:f0:c5:8f
    Device type: Physical Initiator+Target
    Port Index: 5
    Share Area: No
    Device Shared in Other AD: No
    Redirect: No
    Aliases: BlueStor_EMC_FABA
```

3. Use the **wwn** command for Fabric A DCX95 to obtain the Encryption WWNN:

```
DCX95:admin> wwn
10:00:00:05:1e:46:08:00
```

4. Create a new CTC for the Blue Storage 1 using slot 4 and add initiator Red Host HBA 1:

```
DCX95:admin> cryptocfg --create -container disk SID_0950_16cA
   10:00:00:05:1e:46:08:00 4 50:06:04:8a:d5:f0:c5:8f 50:06:04
   :8a:d5:f0:c5:8f
                            Slot     Local/
      EE Node WWN          Number    Remote
10:00:00:05:1e:46:08:00       4      Local
Operation succeeded.

DCX95:admin> cryptocfg --add -initiator SID_0950_16cA 10:00:00:05:1e:0c:1e:1b
   20:00:00:05:1e:0c:1e:1b
Operation succeeded.
DCX95:admin>
```

In this case, LUN numbers are unknown, so the CTC has to be committed prior to using the **discoverLUN** option:

```
DCX95:admin> cryptocfg --commit
Operation succeeded.

DCX95:admin> cryptocfg --discoverLUN SID_0950_16cA
Container name:          SID_0950_16cA
Number of LUN(s):                4
Host:                    10:00:00:05:1e:0c:1e:1b
LUN number:              0x0
LUN serial number:       60060480000190300950533030303230
Key ID state:            Key ID not available

Host:                    10:00:00:05:1e:0c:1e:1b
LUN number:              0x26
LUN serial number:       60060480000190300950533030304443
Key ID state:            Key ID not available
(Output truncated)
```

5. Add 0x26 as the desired LUN for the CTC.

```
DCX95:admin> cryptocfg --add -LUN SID_0950_16cA 0x26 10:00:00:05:1e:0c:1e:1b
   20:00:00:05:1e:0c:1e:1b
Operation succeeded.
```

6. Issue the **cryptocfg--commit** command:

```
DCX95:admin> cryptocfg --commit
Operation succeeded.
```

7. Verify that the LUN has been successfully added to the new CTC.

   The following output indicates there are now two containers. The initiator Red HBA 1 is in both CTCs: SID_0950_15cB and SID_0950_16cA.

```
DCX95:admin> cryptocfg --show -container -all -stat
Encryption group name:  brocade
Number of Container(s): 2                (Both containers are represented)
```

```
Container name:          SID_0950_16cA   (New containter)
Type:                    disk
EE node:                 10:00:00:05:1e:46:08:00

EE slot:                   4    (Encryption blade in slot 4)
EE hosting container:    current
Target:    50:06:04:8a:d5:f0:c5:8f 50:06:04:8a:d5:f0:c5:8f(Blue Storage 1)
Target PID:              5f0500
VT:                      20:0a:00:05:1e:54:17:49 20:0a:00:05:1e:54:17:49
VT PID:                  5fb401
Number of host(s):       1
Number of rekey session(s):     0
Host:       10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b  (Red Host 1)
Host PID:                5d0800
VI:                      20:0b:00:05:1e:54:17:49 20:0c:00:05:1e:54:17:49
VI PID:                  5fb601
Number of LUN(s):        1
LUN number:               0x26                                    (New LUN)
LUN type:                disk
LUN serial number:       60060480000190300950533030304443
Encryption mode:         cleartext
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Clear text
Encryption algorithm:    None
Key ID state:            Key ID not Applicable

Container name:          SID_0950_15cB         (Original Storage 1 container)
Type:                    disk
EE node:                 10:00:00:05:1e:46:08:00
EE slot:                   12                     (Encryption blade in slot 12)
EE hosting container:    current
Target:          50:06:04:8a:d5:f0:c5:ae 50:06:04:8a:d5:f0:c5:ae (Red
                                                                  Storage 1)
Target PID:              5f0400
VT:                      20:00:00:05:1e:54:17:49 20:00:00:05:1e:54:17:49
VT PID:                  5ff601
Number of host(s):       2
Number of rekey session(s):     0
Host:                      10:00:00:05:1e:0c:1e:1b 20:00:00:05:1e:0c:1e:1b
   (Red HBA1)
Host PID:                5d0800
VI:                      20:01:00:05:1e:54:17:49 20:02:00:05:1e:54:17:49
VI PID:                  5ff801
Number of LUN(s):        3

LUN number:              0x27
LUN type:                disk
LUN serial number:       60060480000190300950533030304535
Encryption mode:         encrypt
```

```
Encryption format:      native
Encrypt existing data:  enabled
Rekey:                  disabled
Internal EE LUN state:  Encryption enabled
Encryption algorithm:   AES256-XTS
Key ID state:           Read write
Key ID:                 bc:c3:ef:b8:55:db:35:2c:5a:c9:ad:9d:c2:09:30:d9
Key creation time:      Mon Jun  8 23:03:21 2009
LUN number:             0x28
LUN type:               disk
LUN serial number:      60060480000190300950533030304536
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable

LUN number:             0x29
LUN type:               disk
LUN serial number:      60060480000190300950533030313134
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable

Host:      21:01:00:1b:32:21:59:1b 20:01:00:1b:32:21:59:1b (Blue Host 1)
Host PID:               5b0900
VI:                     20:08:00:05:1e:54:17:49 20:09:00:05:1e:54:17:49
VI PID:                 62fa01
Number of LUN(s):       1

LUN number:             0x31
LUN type:               disk
LUN serial number:      60060480000190300950533030304087
Encryption mode:        cleartext
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Clear text
Encryption algorithm:   None
Key ID state:           Key ID not Applicable
Operation succeeded.
```

Repeat steps for Red Host HBA 2 creating a CTC on Blue Storage 2 defining LUN 0x26 in Fabric B DCX98.

8. From Group Leader Node Fabric A DCX95, create CTC "SID_0950-1ca" on Blue Storage followed by adding Red Host HBA 2 and defining LUN 0x26.

   The command syntax is as follows:

```
DCX98:root> cryptocfg --create –container <disk | tape> <crypto target container
   name> <<node WWN> [slot number]> <Target WWPN> <Target WWNN>
```

   For example:

```
DCX95:admin> cryptocfg --create -container disk SID_0950-1ca
   10:00:00:05:1e:46:50:00 4 50:06:04:8a:d5:f0:c5:80 50:06:04:8a:d5:f0:c5:80
```

9. Add Initiator Red Host HBA 2:

   The command syntax is as follows:

```
DCX95:admin> cryptocfg --add –initiator <crypto target container name> <<Initiator
   WWPN> <Initiator WWNN>>
```

   For example:

```
DCX95:admin> cryptocfg --add -initiator SID_0950-1ca 10:00:00:05:1e:0c:1e:1c
   20:00:00:05:1e:0c:1e:1c
Operation succeeded
```

10. Add LUN 0x26.

```
DCX95:admin> cryptocfg --add -LUN SID_0950-1ca 0x26 10:00:00:05:1e:0c:1e:1c
   20:00:00:05:1e:0c:1e:1c
Operation succeeded.
```

11. Perform the commit operation:

```
DCX95:admin> cryptocfg --commit
Operation succeeded.
```

Step 5 is now complete and the following final checkpoint reached.

CHECKPOINT 5k:

Red HBA Host 1 has been set up for access to LUN 0x26 on another CTC (SID_0950_16cA), newly created on the slot 4 EE for Blue Storage 1 in Fabric A. Additionally, Red HBA Host 2 has been set up for access to the same LUN 0x26 on another CTC (SID_0950-1cA), newly created on the slot 4 EE for Blue Storage 2 in fabric B.

If you want to encrypt the data on LUN 0x26, modify LUN 0x26 as described in detail in "Step 5: Create CTCs in both Fabrics A and B" on page 192.

Summary of CTCs, hosts and LUNs

Figure 32 shows the final configuration of CTCs, hosts, and LUNs.

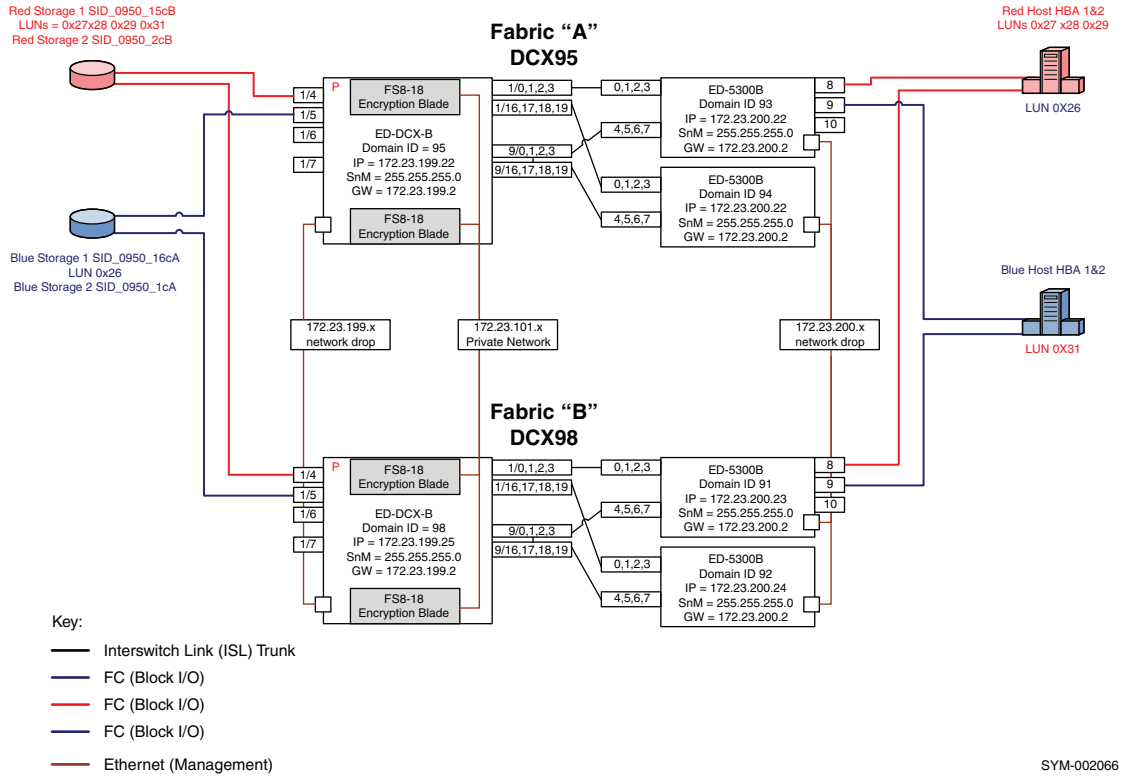


Figure 32 Final configuration of CTCs, hosts, and LUNs

This completes the overview for implementing the EMC/Brocade fabric-based encryption solution.

The overview and terms and concepts, along with the installation of the PB-DCX-16EB Encryption Blades and best practice recommendations, provide the necessary guidance for deployment.

For further assistance, refer to the *EMC Connectrix B Fabric OS Encryption Administrator's Guide*.

TimeFinder case study

This section describes how to set up a Brocade fabric-based encryption solution in an EMC TimeFinder environment. This case study is based on Brocade FOS v6.4.2.

This section contains the following information:

- "Configuration requirements" on page 221
- "Reference architecture" on page 222
- "Summary of initialization steps" on page 223
- "Rekeying encrypted source LUNs in the TimeFinder environment" on page 231

Assumptions

For the steps that follow, this case study assumes the following:

- The Brocade fabric is already properly configured, as detailed in the "Brocade fabric-based encryption case study" on page 164.
- TimeFinder SNAP/CLONE/BCV are already configured.

Configuration requirements

The configuration requirements for TimeFinder encryption solutions are as follows:

- Encryption for EMC TimeFinder can only be done with RSA Key Manager (RKM).
- Replication feature needs to be enabled.

Note: If replication is enabled, firmware downgrades to older FOS releases will be disallowed until the replication feature is disabled. The replication feature cannot be disabled if there are LUNs in the encryption group (EG) configured with the **–newLUN** option.

Reference architecture

The reference architecture for this case study is shown in Figure 33.

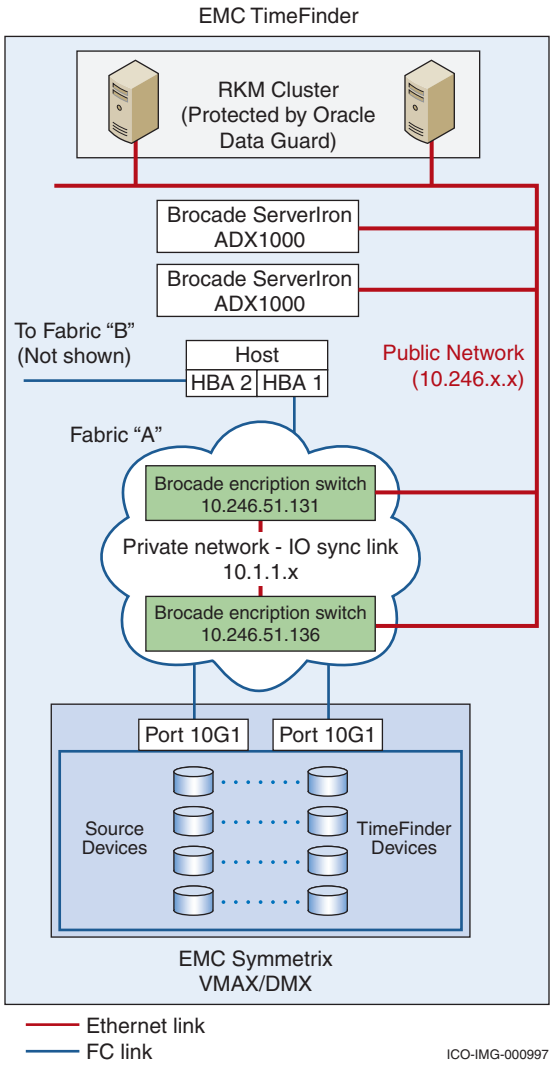


Figure 33 TimeFinder case study architecture

The architecture shown in Figure 33 consists of Fabric A and Fabric B. This case study provides the step-by-step instructions for Fabric A.

The setup steps for Fabric B should be nearly identical to the steps involved to setup Fabric A. Fabric A consists of two Brocade encryption switches called Mace131 and Mace136. These two switches are made up of a single Encryption Group (EG), as shown in this architecture.

Summary of initialization steps

The following is a high-level overview of the steps needed to configure Brocade fabric-based encryption in an EMC TimeFinder environment. Each step is explained in more detail in this section.

- "Step 1: Ensuring that both fabrics are set for defzone--noaccess" on page 223
- "Step 2: Enabling remote application mode" on page 224
- "Step 3: Creating the target Crypto Target Container (CTC)" on page 224
- "Step 4: Adding the LUNs to the CTC" on page 227

Step 1: Ensuring that both fabrics are set for defzone--noaccess

Before committing any encryption configurations in a fabric, the default zoning for that fabric must be set to **No Access**. The **No Access** setting ensures that no two devices on the fabric can communicate with one another without going through a regular zone or a redirection zone.

Zoning is set to **All Access** by default. To ensure the default zoning is set to **No Access**, complete the following steps:

1. Issue the following command on all fabrics to check the zoning setting.

```
i2051131:admin> defzone --show
Default Zone Access Mode
committed - All Access
transaction - No Transaction
```

2. If the default zoning is set to **All Access**, change it to **No Access** by issuing the following command from the primary FCS switch in that fabric:

```
i2051131:admin> defzone --noaccess
i2051131:admin> cfgfsave
The change will be applied within the entire fabric.
```

3. Verify that the the defzone is set to **No Access** by issuing the following command.

```
i2051131:admin> defzone --show
Default Zone Access Mode
committed - No Access
transaction - No Transaction
```

Repeat these steps for each fabric.

Step 2: Enabling remote application mode

The remote replication features are supported in Brocade FOS v6.4 and later. Remote application is disallowed under the following conditions:

- One of the nodes in an encryption group (EG) running an earlier version of the Brocade FOS.
- A node is downgraded to an earlier version of the Brocade FOS.

To enable the remote replication features of the Brocade encryption solution, issue the **cryptocfg –set –replication enable** command. This mode can be enabled only when the configured key vault is RSA Key Manager.

```
i2051131:admin> cryptocfg --set -replication enable
Set replication mode status: Operation Succeeded.
```

To enable replication mode on Site 2:

```
i2051132:admin> cryptocfg --set -replication enable
Set replication mode status: Operation Succeeded.
```

To verify the replication is enabled, issue the following command:

```
i2051131:admin> cryptocfg --show -groupcfg
Encryption Group Name: R1_131_136
Failback mode: Auto
Replication mode: Enabled
Heartbeat misses: 3
Heartbeat timeout: 2
Key Vault Type: RKM
System Card: Disabled
Output truncated….
```

Step 3: Creating the target Crypto Target Container (CTC)

The Crypto Target Containers (CTC), which will be part of the encrypted traffic flow, must be added to the Encryption Engine (EE). When setting up this solution, you must ensure that the source LUNs and TimeFinder LUNs are owned by an EE.

IMPORTANT

Before configuring the CTCs, ensure that standard zoning is in place for the initiators and storage ports. In this case study, both source and TimeFinder LUNs are mapped to the same Symmetrix storage port. If TimeFinder LUNs are mapped to different Symmetrix storage ports, additional CTCs need to be created so the initiator can access these LUNs.

To create the CTC, from the Encryption Group (EG) leader node, complete the following steps:

1. Find the switch wwn by issuing the following command:

```
i2051131:admin> wwn
10:00:00:05:1e:c1:75:ac
```

2. Find the Target information by issuing the following command:

```
i2051131:admin> nodefind 50:00:09:72:08:2f:45:a4
Local:
 Type Pid    COS     PortName                NodeName                SCR
 N    010100;      3;50:00:09:72:08:2f:45:a4;50:00:09:72:08:2f:44:00; 0x00000003
    FC4s: FCP
    PortSymb: [94] "SYMMETRIX::000192603025::SAF-10gA::FC::5875_212+::EMUL
   B80F0000 376FEA87 7EB7D8 06.22.11 14:51"
    NodeSymb: [38] "SYMMETRIX::000192603025::FC::5875_212+"
    Fabric Port Name: 20:01:00:05:1e:c1:75:ac
    Permanent Port Name: 50:00:09:72:08:2f:45:a4
    Device type: Physical Target
    Port Index: 1
    Share Area: No
    Device Shared in Other AD: No
    Redirect: Yes target
    Partial: No
    Aliases:
```

3. Create a Crypto Target Container by issuing the following command:

```
i2051131:admin> cryptocfg --create -container disk Symm10G0_TF
   10:00:00:05:1e:c1:75:ac 50:00:09:72:08:2f:45:a4 50:00:09:72:08:2f:44:00
                            Slot     Local/
      EE Node WWN          Number    Remote
10:00:00:05:1e:c1:75:ac       0      Local
Operation succeeded.
```

Note: This operation must be done in the group leader node. If the storage port is connected to a different switch, that switch WWN will be used instead.

The initiator is added to the CTC to allow the discovery of the LUNs behind the storage port to which the added initiator has access. To add the initiator to the CTC:

a. Find the Initiator information by issuing the following command:

```
i2051131:admin> nodefind 10:00:00:00:c9:74:92:e4
Local:
 Type Pid    COS     PortName                NodeName                  SCR
 N    010200;    2,3;10:00:00:00:c9:74:92:e4;20:00:00:00:c9:74:92:e4; 0x00000003
    FC4s: FCP
    PortSymb: [34] "Emulex PPN-10:00:00:00:c9:74:92:e4"
    NodeSymb: [40] "Emulex LPe12002-E FV1.10A5 DV8.2.0.48.2p"
    Fabric Port Name: 20:02:00:05:1e:c1:75:ac
    Permanent Port Name: 10:00:00:00:c9:74:92:e4
    Device type: Physical Initiator
    Port Index: 2
    Share Area: No
    Device Shared in Other AD: No
    Redirect: No
    Partial: No
    Aliases:
```

b. Add the initiator to the CTC by issuing the following command:

```
i2051131:admin> cryptocfg --add -initiator Symm10G0_TF 10:00:00:00:c9:74:92:e4
   20:00:00:00:c9:74:92:e4
Operation succeeded.
```

Once the commit operation is performed (shown next), all I/O to all LUNs behind the configured storage port will be stopped or failed since the commit operation will cause Frame Redirection to occur in the Fabric. Any I/Os to the configured storage port will now be routed though the EE. At this point, the CTC does not contain any actual LUNs; therefore I/O will be stopped or failed. The commit operation can allow up to 25 commit transactions.

```
i2051131:admin> cryptocfg --commit
Operation succeeded.
```

c. Show the CTC by issuing the following command:

```
i2051131:root> cryptocfg --show -container -all -cfg
Encryption group name:  R1_131_136
Number of Container(s): 1

Container name:         Symm10G0_TF
Type:                   disk
EE node:                10:00:00:05:1e:c1:75:ac
EE slot:                0
```

```
Target:                  50:00:09:72:08:2f:45:a4 50:00:09:72:08:2f:44:00
VT:                      20:00:00:05:1e:54:22:40 20:01:00:05:1e:54:22:40
Number of host(s):       1
Configuration status:    committed
Host:                    10:00:00:00:c9:74:92:e4 20:00:00:00:c9:74:92:e4
VI:                      20:02:00:05:1e:54:22:40 20:03:00:05:1e:54:22:40
Number of LUN(s):        0
Operation succeeded.
```

Step 4: Adding the LUNs to the CTC

There are two possible LUN configuration scenarios to consider when deploying Brocade fabric-based encryption solution in a TimeFinder environment.

- Creating new source LUNs that can later be replicated, snapped, or cloned.
- Migrating data from existing encrypted or cleartext source LUNs to new source LUNs that can be replicated, snapped, or cloned.

Note: EMC PowerPath Migration is recommended for migration of existing data from the existing source LUN to the new source LUN.

This case study will describe the first scenario, *creating new source LUNs that can later be replicated, snapped, or cloned*.

The following restrictions must be followed for TimeFinder to correctly work with the Brocade fabric-based encryption solution:

- Source and TimeFinder LUNs must be of the same size.
- Encryption policies must be consistent for both source and TimeFinder LUN.
- Once the LUN is added to the CTC using the **–newLUN** option, it must not be resized.
- Automatic rekey feature is not allowed with **–newLUN** option.

After completing the steps in "Step 3: Creating the target Crypto Target Container (CTC)" beginning on page 224, the next step is to add the LUNs to the CTC. To add the LUNs to the CTC, complete the following steps:

1. Discover the LUNs behind the storage port.

   The following output shows how LUNs are discovered and displayed.

```
i2051131:root> cryptocfg --discoverLUN Symm10G0_TF
Container name:          Symm10G0_TF
```

```
Number of LUN(s):               24
Host:                   10:00:00:00:c9:74:92:e4
LUN number:             0x0
LUN serial number:      60000970000192603025533030343841
LUN connectivity state: Connected
Key ID state:           Key ID not available

Host:                   10:00:00:00:c9:74:92:e4
LUN number:             0x1
LUN serial number:      60000970000192603025533030343842
LUN connectivity state: Connected
Key ID state:           Key ID not available

Output truncated.

Host:                   10:00:00:00:c9:74:92:e4
LUN number:             0x9
LUN serial number:      60000970000192603025533030353937
LUN connectivity state: Connected
Key ID state:           Read write
Key ID:                 Key ID not available

Output truncated.

Host:                   10:00:00:00:c9:74:92:e4
LUN number:             0x11
LUN serial number:      60000970000192603025533030373131
LUN connectivity state: Connected
Key ID state:           Key ID not available

Operation succeeded.
```

Note: If the source LUNs are not empty and therefore contain customer data, then these LUNs need to be migrated to the large LUNs to allow room for encryption metadata.

The process includes creating new or additional LUNs that are larger by 3 blocks, adding those LUNs with the **–newLUN** option to the same CTCs, and using PowerPath Migration Enabler (PPME) to migrate data from the existing LUN to the larger LUN.

Details of this process is included in *Fabric OS Encryption Administrator's Guide Supporting RSA Key Manager(RKM) Environments, Supporting Fabric OS 6.4.0*, located under the Documents section of MyBrocade located at https://login.brocade.com.

IMPORTANT

If there are multiple paths to the same LUNs, ensure that each path is hosted by an EE and in the same EG. All encryption policies must be consistent among paths to the same LUNs.

In this case study, LUN 0x01 is the source LUN. LUN 0x09 is the TimeFinder SNAPSHOT of LUN 0x01. LUN 0x11 is the BCV LUN that is paired with the source LUN 0x01.

2. Add source LUN 0x01:

```
i2051131:root> cryptocfg --add -LUN Symm10G0_TF 0x1 10:00:00:00:c9:74:92:e4
   20:00:00:00:c9:74:92:e4 -lunstate cleartext -encrypt -newLUN
Adding LUN with '-newLUN' option will render last 3 blocks on the LUN unusable
ARE YOU SURE  (yes, y, no, n): [no] y
Operation succeeded.

Commit the change:

i2051131:root> cryptocfg --commit
Operation succeeded.
```

IMPORTANT

The above command assumes that there is no existing or valid user data on the LUN. Therefore, this will destroy any existing user data on the LUN since the default option is disable_encexistingdata, resulting in any existing data on the LUN being lost.

If the LUN had existing data, migration to the larger LUN is required to accommodate the Brocade secondary metadata and maintain the integrity of the existing user data.

A detailed explanation of the migration process can be found in *Fabric OS Encryption Administrator's Guide Supporting RSA Key Manager (RKM) Environment, Supporting Fabric OS v6.4.0* at *https://login.brocade.com*.

3. Add TimeFinder/SNAP LUN 0x09:

```
i2051131:root> cryptocfg --add -LUN Symm10G0_TF 0x9 10:00:00:00:c9:74:92:e4
   20:00:00:00:c9:74:92:e4 -lunstate encrypted -encrypt -newLUN
Operation succeeded.
```

4. Add BCV LUN 0x11:

```
i2051131:root> cryptocfg --add -LUN Symm10G0_TF 0x11 10:00:00:00:c9:74:92:e4
   20:00:00:00:c9:74:92:e4 -lunstate encrypted -encrypt -newLUN
Operation succeeded.

i2051131:root> cryptocfg --commit
Operation succeeded.
```

IMPORTANT

When adding a TimeFinder LUN into the CTC, ensure that the initial LUN state is encrypted, as opposed to cleartext, when adding the source LUN.

5. Verify that all LUNs are properly added into the CTC:

```
i2051131:root> cryptocfg --show -container Symm10G0_TF  -stat

Container name:          Symm10G0_TF
Type:                    disk
EE node:                 10:00:00:05:1e:c1:75:ac
EE slot:                 0
EE hosting container:    current
Target:                  50:00:09:72:08:2f:45:a4 50:00:09:72:08:2f:44:00
Target PID:              010100
VT:                      20:00:00:05:1e:54:22:40 20:01:00:05:1e:54:22:40
VT PID:                  012001
Number of host(s):       1
Number of rekey session(s):     0
Host:                    10:00:00:00:c9:74:92:e4 20:00:00:00:c9:74:92:e4
Host PID:                010200
VI:                      20:02:00:05:1e:54:22:40 20:03:00:05:1e:54:22:40
VI PID:                  012401
Number of LUN(s):        3
LUN number:              0x1
LUN type:                disk
LUN serial number:       60000970000192603025533030343842
Encryption mode:         encrypt
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Encryption enabled
Encryption algorithm:    AES256-XTS
Key ID state:            Read write
New LUN:                 Yes
Replication LUN type:    Primary
Key ID:                  aa:c3:6c:ca:a3:dc:72:50:6f:89:18:33:b3:78:4c:de
Key creation time:       Fri Jun 24 18:29:21 2011
LUN number:              0x9
LUN type:                disk
```

```
LUN serial number:      60000970000192603025533030353937
Encryption mode:        encrypt
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Encryption enabled
Encryption algorithm:   AES256-XTS
Key ID state:           Read write
New LUN:                Yes
Replication LUN type:   Mirror
Key ID:                 aa:c3:6c:ca:a3:dc:72:50:6f:89:18:33:b3:78:4c:de
Key creation time:      Fri Jun 24 18:29:21 2011
LUN number:             0x11
LUN type:               disk
LUN serial number:      60000970000192603025533030373131
Encryption mode:        encrypt
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  LUN setup
Encryption algorithm:   AES256-XTS
Key ID state:           Unknown
New LUN:                Yes
Replication LUN type:   Mirror
Key ID:                 aa:c3:6c:ca:a3:dc:72:50:6f:89:18:33:b3:78:4c:de
Operation succeeded.
```

Rekeying encrypted source LUNs in the TimeFinder environment

Note: A detailed explaination of the rekeying process for SRDF/TimeFinder LUNs can be found in the *Fabric OS Encryption Administrator's Guide Supporting RSA Key Manager(RKM) Environments, Supporting Fabric OS 6.4.0*, located under the Documents section of MyBrocade located at https://login.brocade.com.

Consider the following when rekeying encrypted source LUNs in the TimeFinder environment.

- Auto rekey is disabled for source and TimeFinder LUNs.
- Manual rekey works only on the source/primary LUNs. Rekey TimeFinder/mirror LUNs requires **–include_mirror** option.
- **Cryptocfg –manual_rekey –all –include_mirror** will rekey all the source/primary and TimeFinder/mirror LUNs. If this action is needed, ensure all TimeFinder/mirror LUNs are read and write enabled.

SRDF encryption case study

This case study describes how to deploy Brocade fabric-based encryption in a two-site configuration in which EMC SRDF is utilized to replicate encrypted data between the two sites.

This section contains the following information:

- "Configuration requirements" on page 233
- "Reference architecture" on page 234
- "Summary of initialization steps" on page 235
- "Configuring the Encryption Engines" on page 236
- "Configuring the Encryption Groups" on page 242
- "Configuring the High Availability (HA) clusters" on page 246
- "Setting up RKM key vault" on page 248
- "Setting up ADX IPLB (IP Load Balance)" on page 256
- "Registering the RKM KV on the Encryption Group leader" on page 262
- "Dealing with certificate expiration (KAC, Apache Server, and CA)" on page 266
- "Generating and backing up the EG Master key" on page 272
- "Ensuring that both fabrics are set for defzone--noaccess" on page 276
- "Enabling remote replication mode" on page 277
- "Creating the CTCs at each site" on page 278
- "Adding the source SRDF LUNs to each CTC at Site 1" on page 280
- "Adding the source SRDF LUNs to each CTC at Site 2" on page 284

Configuration requirements

The following are configuration requirements for SRDF encryption solutions:

- Encryption SRDF LUNs can be done only when the key vault configured is the RSA RKM
- For SRDF it is assumed that there is a clustered pair of RKMs at the local site and a clustered pair of RKMs at the remote site. The clustered pairs must then be configured as part of the same RKM group

Note: If replication is enabled, firmware downgrades to older FOS releases will be disallowed until the replication feature is disabled. The replication feature cannot be disabled if there are LUNs in the Encryption Group (EG) configured with the **-newLUN** option.

Reference architecture

The reference architecture for this case study is shown Figure 34.

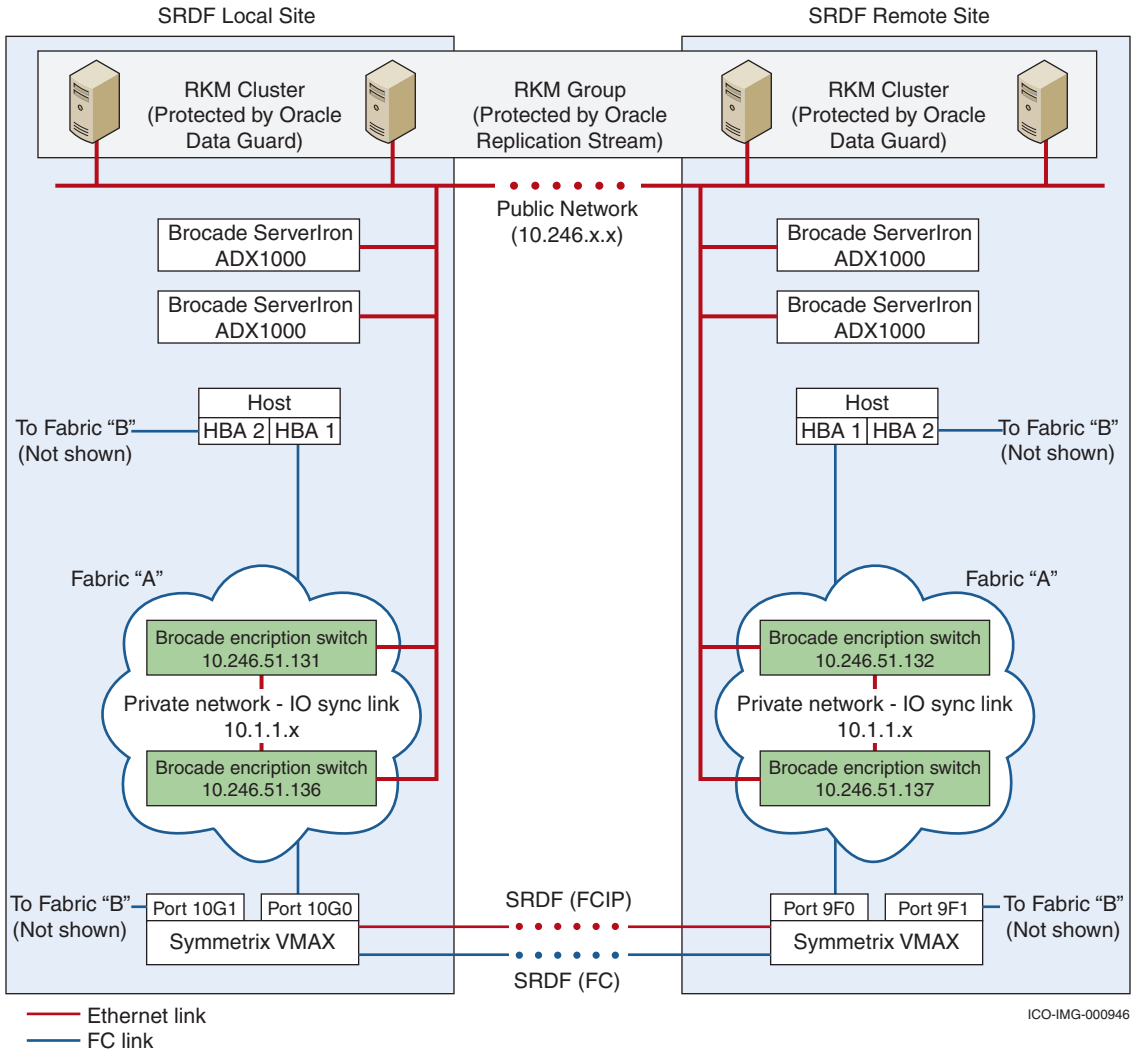


Figure 34 Encryption for two sites with SRDF

Note that each of the two sites in the architecture consists of both FabricA and FabricB. For example, FabricA at Site1 consists of two Brocade Encryption switches called Mace131 and Mace136. Likewise, FabricA at Site2 consists of two Brocade Encryption switches called

Mace132 and Mace137. Each site will be made up of a single Encryption Group (EG) as shown in the architecture.

For ease of explanation, the configuration steps in the following section will only provide step-by-step instructions for setting up FabricA at each site. That is, each site's FabricB setup steps will not be addressed. The setup steps for each FabricB would be nearly identical to the steps involved for setting up FabricA. Any differences will be discussed.

Summary of initialization steps

IMPORTANT

Setting up Brocade fabric-based encryption requires initialization steps, which are performed only once and which must be executed in the specified order indicated.

The most challenging aspect of setting up an encryption solution involves the initial configuration of the Brocade encryption switches, RKM KVs, and the IPLBs. Once communication has been established between all encryption switches in an EG and their associated RKM KVs, the majority of the setup work has been completed.

The final steps of an encryption setup consisting of configuring storage targets (CTCs) and their associated LUNs are straightforward and take far less time.

Assumptions

For the steps that follow, it is assumed all of the encryption switches, and RKM KVs, and IPLBs have been powered on. It is further assumed that all of the FC cabling, management interface cabling, and I/O Sync Link cabling between encryption engines have been completed.

The following is a high-level overview of the configuration process. Each step is explained in more detail in the next section, "Configuring the Encryption Engines."

- "Step 1: Ensure that your FOS version is correct" on page 236
- "Step 2: Initialize each of the encryption nodes" on page 236
- "Step 3: Initialize each of the Encryption Engines" on page 238
- "Step 4: Register each of the Encryption Engines" on page 239
- "Step 5: Enabling the Encryption Engines" on page 240
- "Step 6: Configure each of the Encryption Engine's cluster link interfaces" on page 240

Configuring the Encryption Engines

Complete the following steps to configure the Encryption Engines.

Step 1: Ensure that your FOS version is correct

Log in to each of the EG nodes and use the following command to ensure that your FOS version is up to date:

```
i2051131:admin> firmwareshow
Appl     Primary/Secondary Versions
------------------------------------------
FOS      v6.4.1a
         v6.4.1a
```

Step 2: Initialize each of the encryption nodes

All the Brocade encryption switches need to be initialized prior to use. To initialize an EG node, the **cryptocfg --initnode** command is used, which generates the following security parameters and certificates:

- Node CP certificate
  - This is created during node initialization (cryptocfg --initnode), exchanged with the group leader, and used for authenticating a member node with the group leader.

    Note: A group leader is the first node created in an encryption group that acts as a group and cluster manager. The group leader manages and distributes all group-wide and cluster-wide configurations to all members of the group or cluster. If the group leader node is power cycled, another group member becomes the group leader. When the previous group leader comes back online, it becomes a group member.

- Authentication Center (KAC) certificate *or* KAC CSR (Certificate Signing Request)
  - The KAC certificate is a self-signed certificate that could be used for authentication with the RSA Key Manager (RKM) key vault.
  - The CSR is a signing request certificate that would need to be signed by a Certificate Signing Authority.
- FIPS Crypto Officer
  - This is used for internal authentication between processors. This certificate is exchanged internally and therefore requires no manual intervention.

- FIPS user
  - Like the FIPS Crypto Officer, this certificate is also used for internal authentication between processors and is exchanged internally, so requires no manual intervention.

IMPORTANT

Once your EG has been set up and made operational, there is never a need to reissue the cryptocfg --initNode command. Issuing a cryptocfg –initNode command on a Node after it has been made operational will result in that Node no longer being able to communicate to the EG Leader or the RKM KVs (effectively rendering the Node inoperable until it is reconfigured).

Perform the following steps on each of the Brocade Encryption switches in all your fabrics:

```
i2051132:admin> cryptocfg --initnode
This will overwrite all identification and authentication data
ARE YOU SURE  (yes, y, no, n): [no] y
Operation succeeded.
```

- Before **cryptocfg –initNode:**

```
i2051132:admin> cryptocfg --show -localEE

    EE Slot:0
        SP state:Waiting for initnode
        EE key status not available: SP TLS connection is not up.
    No HA cluster membership
    EE Attributes:
    Link IP Addr        : 0.0.0.0
    Link GW IP Addr     : 0.0.0.0
    Link Net Mask       : 0.0.0.0
    Link MAC Addr       :
    Link MTU            : 0
    Link State          : DOWN
    Media Type          : MEDIA NOT DEFINED
    Rebalance Recommended: NO
    System Card           : Disabled
    System Card Label   :
    System Card CID     :

Remote EE Reachability :
```

- After **cryptocfg –initNode:**

```
i2051132:admin> cryptocfg --show -localEE

    EE Slot:0
```

```
        SP state:Waiting for initEE
        EE key status not available: SP TLS connection is not up.
    No HA cluster membership
    EE Attributes:
    Link IP Addr        : 0.0.0.0
    Link GW IP Addr     : 0.0.0.0
    Link Net Mask       : 0.0.0.0
    Link MAC Addr       :
    Link MTU            : 0
    Link State          : DOWN
    Media Type          : MEDIA NOT DEFINED
    Rebalance Recommended: NO
    System Card           : Disabled
    System Card Label   :
    System Card CID     :

Remote EE Reachability :
```

Step 3: Initialize each of the Encryption Engines

Each of the EEs must be initialized before they can be utilized for encryption purposes. Initializing the EE has the effect of generating critical security parameters and certificates in the crypto module.

Perform the following steps on each of the EEs in all of your fabrics:

```
i2051132:admin> cryptocfg --initEE
This will overwrite previously generated identification and authentication data
ARE YOU SURE  (yes, y, no, n): [no] y
Operation succeeded.
```

After **cryptocfg –initEE**:

```
i2051132:admin> cryptocfg --show -localEE

    EE Slot:0
        SP state:Waiting for regEE
        EE key status not available: SP TLS connection is not up.
    No HA cluster membership
    EE Attributes:
    Link IP Addr        : 0.0.0.0
    Link GW IP Addr     : 0.0.0.0
    Link Net Mask       : 0.0.0.0
    Link MAC Addr       :
    Link MTU            : 1500
    Link State          : UP
    Media Type          : MEDIA NOT DEFINED
    Rebalance Recommended: NO
    System Card           : Disabled
    System Card Label   :
    System Card CID     :

Remote EE Reachability :
```

Note: If this Encryption Engine was previously utilized for encryption purposes, the **initEE** command will fail with the following message "Operation failed: Encryption Engine (EE) not zeroized". If you receive this error message, you will need to issue the **cryptocfg –zeroizeEE** command. As part of this command output and confirmation, you will receive a warning that the encryption switch will be rebooted as part of the zeroization process. Once rebooted, issuing the **initEE** command will be successful.

Step 4: Register each of the Encryption Engines

Each of the EEs must be registered before they can be utilized for encryption purposes. Registering the EE forces authentication between the crypto-module and the Encryption Node's control processor (CP), using the FIPs User and FIPs Crypto Officer certificates previously mentioned.

Perform the following steps on each of the EEs in all of your fabrics:

```
i2051132:admin> cryptocfg --regEE
Operation succeeded.
```

After **cryptocfg –regEE**:

```
i2051132:admin> cryptocfg --show -localEE
    EE Slot:0
        SP state:Waiting for enableEE
Key vault type not set
    No HA cluster membership
    EE Attributes:
    Link IP Addr        : 0.0.0.0
    Link GW IP Addr     : 0.0.0.0
    Link Net Mask       : 0.0.0.0
    Link MAC Addr       :
    Link MTU            : 1500
    Link State          : UP
    Media Type          : MEDIA NOT DEFINED
    Rebalance Recommended: NO
    System Card           : Disabled
    System Card Label   :
    System Card CID     :

Remote EE Reachability :
```

Step 5: Enabling the Encryption Engines

Each of the EEs must be enabled before they can be used.

Note: Every time a Brocade Encryption Switch or ED-DCX-B or DCX-4S chassis containing one or more PB-DCX-16EB blades goes through power cycle event, or after issuing **slotpoweroff** <*slot number*> followed by **slotpoweron** <*slot number*> for a PB-DCX-16EB blade in ED-DCX-B or DCX-4S chassis, the EE must be enabled manually. Hosts cannot access the storage LUNs through the storage paths exposed on this EE until it is enabled.

Perform the following steps on each of the EEs in all your fabrics:

```
i2051132:admin> cryptocfg --enableEE
Operation succeeded.
```

After **cryptocfg -enableEE**

```
i2051132:admin> cryptocfg --show -localEE

    EE Slot:0
        SP state:Operational; Need Valid KEK
    Current Master KeyID:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
    Alternate Master KeyID:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
    No HA cluster membership
    EE Attributes:
    Link IP Addr        : 0.0.0.0
    Link GW IP Addr     : 0.0.0.0
    Link Net Mask       : 0.0.0.0
    Link MAC Addr       :
    Link MTU            : 1500
    Link State          : UP
    Media Type          : MEDIA NOT DEFINED
    Rebalance Recommended: NO
    System Card           : Disabled
    System Card Label   :
    System Card CID     :

Remote EE Reachability :
No info available; No Encryption Group Created
```

Step 6: Configure each of the Encryption Engine's cluster link interfaces

Each of the EEs must be enabled before they can be used.

Each encryption switch or FS8-18 blade has two-gigabit Ethernet ports labeled Ge0 and Ge1. The Ge0 and Ge1 ports connect encryption switches and PB-DCX-16EB blades to other encryption switches and blades. These two ports are bonded together as a single

virtual network interface; therefore, only one IP address is used. The ports provide link layer redundancy, and are collectively referred to as the Cluster Link.

All encryption switches or blades in an EG must be interconnected by their cluster links through a dedicated LAN. Both ports of each encryption switch or blade must be connected to the same IP network, and the same subnet. Static IP addresses should be assigned. VLANs should not be used, and DHCP should not be used.

Without having the Cluster links set up properly, encryption of LUNs would not be possible by the EG.

Perform the following steps on each of the EEs in all your fabrics specifying the correct IP address and subnet mask for each encryption engine:

```
i2051132:admin> ipaddrset -eth0 --add 10.1.1.133/24
IP address is being changed...Done.
```

- Before **ipaddrset:**

```
i2051132:admin> ipaddrshow

SWITCH
Ethernet IP Address: 10.246.51.132
Ethernet Subnetmask: 255.255.255.0
Gateway IP Address: 10.246.51.1
DHCP: Off
eth0: none/none
eth1: none/none
Gateway: none
IPv6 Autoconfiguration Enabled: Yes
Local IPv6 Addresses:
IPv6 Gateways:
```

- After **ipaddrset:**

```
i2051132:admin> ipaddrshow

SWITCH
Ethernet IP Address: 10.246.51.132
Ethernet Subnetmask: 255.255.255.0
Gateway IP Address: 10.246.51.1
DHCP: Off
eth0: 10.1.1.133/24
eth1: none/none
Gateway: none
IPv6 Autoconfiguration Enabled: Yes
Local IPv6 Addresses:
IPv6 Gateways:
```

Configuring the Encryption Groups

Complete the following steps to configure the Encryption Groups (EG).

Step 1: Create each site's Encryption Group

At this point, the individual encryption switches have been brought up, initialized, and enabled for encryption. For each site, the Encryption Groups must now be created.

Upon creating each EG, the EG group leader will automatically be designated.

Note: A group leader is the first node created in an encryption group that acts as a group and cluster manager. The group leader manages and distributes all group-wide and cluster-wide configurations to all members of the group or cluster. If the group leader node is power cycled, another group member becomes the group leader. When the previous group leader comes back online, it becomes a group member.

The following commands create the two encryption groups named R1_131_136 for the local R1 site and R2_132_137 for the remote R2 site:

```
i2051131:admin>cryptocfg --create -encgroup R1_131_136
Encryption group create status: Operation Succeeded.

i2051132:admin>cryptocfg --create -encgroup R2_132_137
Encryption group create status: Operation Succeeded.
```

After the EG **create:**

```
i2051132:admin> cryptocfg --show -groupcfg
Encryption Group Name:R2_132_137
    Failback mode:Auto
    Replication mode:Disabled
    Heartbeat misses:3
    Heartbeat timeout:2
    Key Vault Type:Not Set
    System Card:Disabled

Primary Key Vault not configured
Secondary Key Vault not configured

Authentication Quorum Size:0
Authentication Cards not configured

NODE LIST
Total Number of defined nodes:1
```

```
Group Leader Node Name:10:00:00:05:1e:55:88:b7
Encryption Group state:CLUSTER_STATE_CONVERGED
Crypto Device Config state:In Sync
Encryption Group Config state:In Sync

      Node Name           IP address      Role
10:00:00:05:1e:55:88:b7   10.246.51.132   GroupLeader  (current node)
    EE Slot:0
    SP state:Operational; Not Online
```

Step 2: Set each site's EG key vault type to RKM

Each EG has to have its key vault type set. EMC only supports the encryption solution with RKM KVs. Use the **cryptocfg --set –keyvault** command, shown next, to set both EG KV types to RKM.

- For the R1_131_136 EG from the group leader Node:

```
i2051131:admin> cryptocfg --set -keyvault RKM
Set key vault status: Operation Succeeded.
```

- For the R2_132_137 EG from the group leader Node:

```
i2051132:admin> cryptocfg --set -keyvault RKM
Set key vault status: Operation Succeeded.
```

- After setting the KV type:

```
i2051132:admin> cryptocfg --show -groupcfg
Encryption Group Name:R2_132_137
    Failback mode:Auto
    Replication mode:Disabled
    Heartbeat misses:3
    Heartbeat timeout:2
    Key Vault Type:RKM
    System Card:Disabled

Primary Key Vault not configured
Secondary Key Vault not configured

Additional Key Vault/Cluster Information:
Additional configuration parameters not available

Output truncated.....
```

Step 3: Exchange and register member node CP certificates

Before encryption group member nodes will be allowed to join an EG, they must first provide authentication through the use of Control Processor (CP) certificates. The process involves exporting the CP certificate from every member node of the EG, followed by the import of those same CP certificates by the EG leader of the EG.

Typically, the CP certificates are exported to an SCP-capable host and subsequently imported from the same SCP-capable host to the EG leader node. In the following examples, the SCP-capable host is a Linux server with IP address 10.246.51.50. The commands show the exportation of each member node's (Mace136 from site R1 and Mace137 from site R2) CP certificates:

- From Site R1's member node Mace136:

```
i2051136:admin> cryptocfg --export -scp -CPcert 10.246.51.50 root
   /tmp/136_cpcert.pem
root@10.246.51.50's password:
Operation succeeded.
```

- From Site R2's member node Mace137:

```
i2051136:admin> cryptocfg --export -scp -CPcert 10.246.51.50 root
   /tmp/137_cpcert.pem
root@10.246.51.50's password:
Operation succeeded.
```

The following commands show the importation of each member node's (Mace136 from site R1 and Mace137 from site R2) CP certificates by their respective EG group leaders:

- For Site R1's group leader node Mace131:

```
i2051131:admin> cryptocfg --import -scp 136_cpcert.p12 10.246.51.50 root
   /tmp/136_cpcert.pem
Available Space:4096
Make sure your file size is not greater than 4096.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to proceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.246.51.50's password:
Operation succeeded.
```

- For Site R2's group leader node Mace132:

```
i2051132:admin>  cryptocfg --import -scp 137_cpcert.p12 10.246.51.50 root
   /tmp/137_cpcert.pem
Available Space:24576
Make sure your file size is not greater than 24576.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to proceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.246.51.50's password:
Operation succeeded.
```

Step 4: Add member nodes to each EG

Once the EG leader has imported the member node's CP certificate, it can then register the member node completing the addition of the member node to the EG.

- From Site R1's group leader node Mace131, use the following command to register member node Mace136 with IP address 10.246.51.136 and WWN 10:00:00:05:1e:54:22:75:

```
i2051131:admin> cryptocfg --reg -membernode 10:00:00:05:1e:54:22:75
   136_cpcert.p12 10.246.51.136
Operation succeeded
.
```

- From Site R2's group leader node Mace132, use the following command to register member node Mace137 with IP address 10.246.51.137 and WWN 10:00:00:05:1e:54:22:44:

```
i2051132:admin> cryptocfg --reg -membernode 10:00:00:05:1e:54:22:44
   137_cpcert.p12 10.246.51.137
Operation succeeded.
```

- Verify that the member node has successfully joined the EG by checking for a *Encryption Group state* of 'CLUSTER_STATE_CONVERGED' as shown next:

```
i2051132:admin> cryptocfg --show -groupcfg
Encryption Group Name:R2_132_137
    Failback mode:Auto
    Replication mode:Disabled
    Heartbeat misses:3
    Heartbeat timeout:2
    Key Vault Type:RKM
    System Card:Disabled

Primary Key Vault not configured
Secondary Key Vault not configured

Authentication Quorum Size:0
Authentication Cards not configured


NODE LIST
Total Number of defined nodes:2
Group Leader Node Name:10:00:00:05:1e:55:88:b7
Encryption Group state:CLUSTER_STATE_CONVERGED
Crypto Device Config state:In Sync
Encryption Group Config state:In Sync

      Node Name IP address   Role
10:00:00:05:1e:55:88:b7  10.246.51.132  GroupLeader  (current node)
    EE Slot:0
    SP state:Operational; Not Online
```

```
10:00:00:05:1e:54:22:44  10.246.51.137  MemberNode
    EE Slot:0
    SP state:Operational; Need Valid KEK
```

Configuring the High Availability (HA) clusters

An HA cluster consists of two encryption engines configured to host the same CryptoTargets and to provide Active/Standby failover and failback capabilities in a single fabric. Failover is automatic (not configurable). Failback occurs automatically by default, but is configurable with a manual failback option. All encryption engines in an encryption group share the same DEK for a disk or tape LUN.

An HA cluster has the following limitations:

- The encryption engines that are part of an HA cluster must belong to the same encryption group and be part of the same fabric.
- An HA cluster cannot span fabrics and it cannot provide failover/failback capability within a fabric transparent to host MPIO software.
- Two PB-DCX-16B blades comprising an HA cluster cannot be installed in the same ED-DCX-B chassis.
- Configuration changes must be committed before they take effect. Any operation related to an HA cluster that is performed without a commit operation will not survive across switch reboots, power cycles, CP failover, or HA reboots.
- It is recommended that the HA cluster configuration be completed before you configure storage devices for encryption.

Note: The CLI does not allow creation of an HA cluster if the node is not in the encryption group.

Complete the following step to configure the High Availability clusters.

Create the HA clusters

Each site is comprised of two fabrics—Fabric A and Fabric B. Again, this reference architecture is only showing the configuration steps necessary to configure each site's Fabric A. Therefore, the two encryption engines comprising each site's Fabric A (Mace131 and

Mace136 at Site R1 and Mace132 and Mace137 at Site R2) will be clustered together.

The use of HA clusters is strictly optional for high-availability purposes and was chosen for this reference architecture largely to illustrate the command necessary to set them up.

- For the R1_131_136 EG from the group leader Node:

```
i2051131:admin> cryptocfg --create hacluster R1_131_136_cluster
   10:00:00:05:1e:c1:75:ac 10:00:00:05:1e:54:22:75

                    Slot     Local/
      EE Node WWN  Number    Remote
10:00:00:05:1e:c1:75:ac     0    Local
                    Slot     Local/
      EE Node WWN  Number    Remote
10:00:00:05:1e:54:22:75     0    Remote
Operation succeeded.

i2051131:admin> cryptocfg --commit
Operation succeeded.
```

- For the R2_132_137 EG from the group leader Node:

```
i2051132:admin> cryptocfg --create -hacluster R2_132_137_cluster
   10:00:00:05:1e:55:88:b7 10:00:00:05:1e:54:22:44

                    Slot     Local/
      EE Node WWN  Number    Remote
10:00:00:05:1e:55:88:b7     0    Local
                    Slot     Local/
      EE Node WWN  Number    Remote
10:00:00:05:1e:54:22:44     0    Remote
Operation succeeded.

i2051132:admin> cryptocfg --commit
Operation succeeded.
```

- After creating the HA cluster at Site 2:

```
i2051132:admin> cryptocfg --show -hacluster all

Encryption Group Name: R2_132_137_cluster
Number of HA Clusters: 1

HA cluster name: R2_132_137_cluster - 2 EE entries
Status: Committed
HAC State: Converged

          WWN  Slot NumberStatus
10:00:00:05:1e:55:88:b7       0     Online
10:00:00:05:1e:54:22:44       0     Online
```

At this point, the HA clusters in the other Fabric B fabrics would typically be created.

Setting up RKM key vault

The base RKM key vault cluster setup will be performed by professional services as part of the encryption installation process. This setup includes the following major steps:

- Loading of the RKM server software on two cluster members (primary and secondary).
  - Refer to the *EMC Support Matrix* for the latest supported RKM server and service pack software version. As of FOS v6.4.1a, the supported RKM server software was v2.7.1 (v2.7.1 represents server version 2.7 with Service Pack 1).
- Loading the latest supported hot fixes for the two cluster members.
  - Refer to the *EMC Support Matrix* for the latest supported hot fix software version. As of FOS v6.4.1a, the supported RKM server software was hot fix v2.7.1.3 (the '.3' represents the hot fix version number).

Note: Hot fixes are cumulative, meaning that if the '.3' version is loaded, then both '.1' and .'2' are automatically loaded as well.

- Ensuring that the network setup on both RKM servers has been completed.
- Setting up of the XTS and GCM key classes required for Brocade encryption operation.
- Setting up of the Identity Group which will be associated with all members of a given EG.
- Generating and signing of the Certificate Signing Requests (CSRs) for each of the RKM Apache Web servers.
- Access to the organization's root CA's public certificate and its private key must be obtained for certificate signing purposes.

Note: There are many third-party certificate signing authorities (CAs) that can be used. Commonly, instead of using a third-party CA, openssl (which is automatically loaded as part of the RKM server installation process) can be utilized to generate your own CA.

Note: For this SRDF reference architecture, there is a local site R1 and a remote site R2. Each site will have a pair of clustered RKM servers grouped together to form an RKM Group. The RKM Group will utilize Oracle streams to keep the two sites synchronized together; that is, all DEKs generated at either site will automatically be replicated to the other site. The setup and configuration of the RKM Group will be completed by professional services as part of the encryption solution installation.

Once the professional installation service team has configured your RKM servers, in order for the Encryption group member nodes to authenticate and then communicate with the RKM servers, the following steps must be performed:

1. Every EG Node must export a Key vault Appliance Certificate (KAC) CSR.
2. Every EG Node CSR must be signed by the same CA.
3. Every EG Node must import its signed KAC.
4. Every EG Node must register its signed KAC.
5. An identity must be created on the RKM cluster for each of its associated EG Nodes.

Step 1: Export Keyvault Appliance Certificate (KAC) Certificate Signing Requests (CSRs)

Typically, the KAC CSR certificates are exported using SCP directly to the RKM servers where they can be signed by the CA that was used to sign the RKM server's public certificate.

In the examples below, the primary RKM server at Site 1 has IP address 10.32.139.32. The commands below show the exportation of each EG node's KAC CSR certificates to the primary RKM at Site 1.

Note: For this example, the certificate signing process is performed on the RKM (the CA is on the RKM). However, it is a common practice to do certificate signing at the CA, which could be on another server.

Note: The EG Nodes at each site could export their KAC CSRs to the RKMs configured at each site. That is, the Site 1 EG Nodes Mace131 and Mace136 could export their KAC CSRs to the Site 1 RKM cluster and the Site 2 EG Nodes Mace132 and Mace137 could export their KAC CSRs to the Site 2 RKM cluster. For illustration and ease, in this example all KAC CSRs are exported to the same RKM server which is located at Site 1.

- From Site R1's Mace131:

```
i2051131:admin> cryptocfg --export -scp -KACcsr 10.32.139.32 root
   /opt/rsa/certs/mace131.csr

   ### Get FN </etc/fabos/certs/sw0/kac_req.csr> ###
root@10.32.139.32's password:
Operation succeeded.
```

- From Site R1's Mace136:

```
i2051136:admin> cryptocfg --export -scp -KACcsr 10.32.139.32 root
   /opt/rsa/certs/mace136.csr

   ### Get FN </etc/fabos/certs/sw0/kac_req.csr> ###
root@10.32.139.32's password:
Operation succeeded.
```

- From Site R2's Mace132:

```
i2051131:admin> cryptocfg --export -scp -KACcsr 10.32.139.32 root
   /opt/rsa/certs/mace132.csr

   ### Get FN </etc/fabos/certs/sw0/kac_req.csr> ###
root@10.32.139.32's password:
Operation succeeded.
```

- From Site R2's Mace137:

```
i2051131:admin> cryptocfg --export -scp -KACcsr 10.32.139.32 root
   /opt/rsa/certs/mace137.csr

   ### Get FN </etc/fabos/certs/sw0/kac_req.csr> ###
root@10.32.139.32's password:
Operation succeeded.
```

Step 2: Sign every EG Node's KAC CSR

The best way to determine what the CA is for a given RKM server is utilizing the **cat/more** or **vi** commands to look at the /etc/httpd/conf.d/ssl.conf file and search for the entry SSLCACertificateFile as shown next:

```
[root@sgeliop32 certs]#cat /etc/httpd/conf.d/ssl.conf

Partial output shown….

#   Certificate Authority (CA):
#   Set the CA certificate verification path where to find CA
#   certificates for client authentication or alternatively one
#   huge file containing all of them (file must be PEM encoded)
SSLCACertificateFile /etc/ssl/certs/trusted_cas.pem
```

```
#   Client Authentication (Type):
#   Client certificate verification type and depth.  Types are

Output truncated……
```

In case you cannot get into the RKM, you can still verify that the RKM servers are using the correct CA, such that the issuer common name (CN) and serial # of the certificate is the same for all, by using the **openssl** command.

```
openssl x509 -in sgeliopvm20-ca.pem -issuer -noout
issuer= /CN=sgeliopvm20/C=SG/ST=SG/L=SG/O=EMC/emailAddress=sgeliopvm20@emc.com

openssl x509 -in sgeliopvm20-ca.pem -serial -noout
serial=A3FA0CF88316AD2A
```

In the following examples, the CA located on the RKM server is used to sign each of the EG Node's KAC CSR. In the commands the CA utilized is located on the RKM servers in the /etc/ssl/certs directory and is called 'trusted_cas.pem'.

```
[root@sgeliop32 certs]# openssl x509 -sha1 -req -days 1825 -in ./mace131.csr -CA
 /etc/ssl/certs/trusted_cas.pem -CAkey ./sgeliopvm20-ca-key.pem -CAcreateserial
-out ./mace_131_signed.pem
Signature ok
subject=/CN=kac.000000051ec175ac/C=US/ST=CA/L=San Jose/O=BRCD/OU=Technical
   Support
Getting CA Private Key
Enter pass phrase for ./sgeliopvm20-ca-key.pem:

[root@sgeliop32 certs]# openssl x509 -sha1 -req -days 1825 -in ./mace132.csr -CA
 /etc/ssl/certs/trusted_cas.pem -CAkey ./sgeliopvm20-ca-key.pem -CAcreateserial
-out ./mace_132_signed.pem
Signature ok
subject=/CN=kac.000000051e5588b7/C=US/ST=CA/L=San Jose/O=BRCD/OU=Technical
   Support
Getting CA Private Key
Enter pass phrase for ./sgeliopvm20-ca-key.pem:

[root@sgeliop32 certs]# openssl x509 -sha1 -req -days 1825 -in ./mace136.csr -CA
 /etc/ssl/certs/trusted_cas.pem -CAkey ./sgeliopvm20-ca-key.pem -CAcreateserial
-out ./mace_136_signed.pem
Signature ok
subject=/CN=kac.000000051e542275/C=US/ST=CA/L=San Jose/O=BRCD/OU=Technical
   Support
Getting CA Private Key
Enter pass phrase for ./sgeliopvm20-ca-key.pem:

[root@sgeliop32 certs]# openssl x509 -sha1 -req -days 1825 -in ./mace137.csr -CA
 /etc/ssl/certs/trusted_cas.pem -CAkey ./sgeliopvm20-ca-key.pem -CAcreateserial
-out ./mace_137_signed.pem
Signature ok
```

```
subject=/CN=kac.000000051e542244/C=US/ST=CA/L=San Jose/O=BRCD/OU=Technical
   Support
Getting CA Private Key
Enter pass phrase for ./sgeliopvm20-ca-key.pem:
```

Step 3: Import the signed KAC certificates

Each EG Node must now import its signed KAC certificate. In the following examples, the primary RKM server has IP address 10.32.139.32. The commands show the importation by each EG of its signed KAC:

- From Site R1's Mace131:

```
i2051131:root> cryptocfg --import -scp mace131_signed_kac.pem 10.32.139.32 root
   /opt/rsa/certs/mace_131_signed.pem
Available Space:20480
Make sure your file size is not greater than 20480.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to procceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.32.139.32's password:
Operation succeeded.
```

- From Site R1's Mace136:

```
i2051136:root> cryptocfg --import -scp mace136_signed_kac.pem 10.32.139.32 root
   /opt/rsa/certs/mace_136_signed.pem
Available Space:8192
Make sure your file size is not greater than 20480.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to procceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.32.139.32's password:
Operation succeeded.
```

- From Site R2's Mace132:

```
i2051132:root> cryptocfg --import -scp mace132_signed_kac.pem 10.32.139.32 root
   /opt/rsa/certs/mace_132_signed.pem
Available Space:8192
Make sure your file size is not greater than 20480.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to procceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.32.139.32's password:
Operation succeeded.
```

- From Site R2's Mace137:

```
i2051137:root> cryptocfg --import -scp mace137_signed_kac.pem 10.32.139.32 root
   /opt/rsa/certs/mace_137_signed.pem
Available Space:8192
Make sure your file size is not greater than 20480.
```

```
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to procceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.32.139.32's password:
Operation succeeded.
```

Step 4: Register the signed KAC certificates

Each EG Node must now register its signed KAC certificate. The following commands show the registration by each EG of its signed KAC.

- From Site R1's Mace131:

```
i2051131:admin> cryptocfg --reg -KACcert mace131_signed_kac.pem primary
Register KAC status: Operation Succeeded.
```

- From Site R1's Mace136:

```
i2051136:admin> cryptocfg --reg -KACcert mace136_signed_kac.pem primary
Register KAC status: Operation Succeeded.
```

- From Site R2's Mace132:

```
i2051132:admin> cryptocfg --reg -KACcert mace132_signed_kac.pem primary
Register KAC status: Operation Succeeded.
```

- From Site R2's Mace137:

```
i2051137:admin> cryptocfg --reg -KACcert mace137_signed_kac.pem primary
Register KAC status: Operation Succeeded.
```

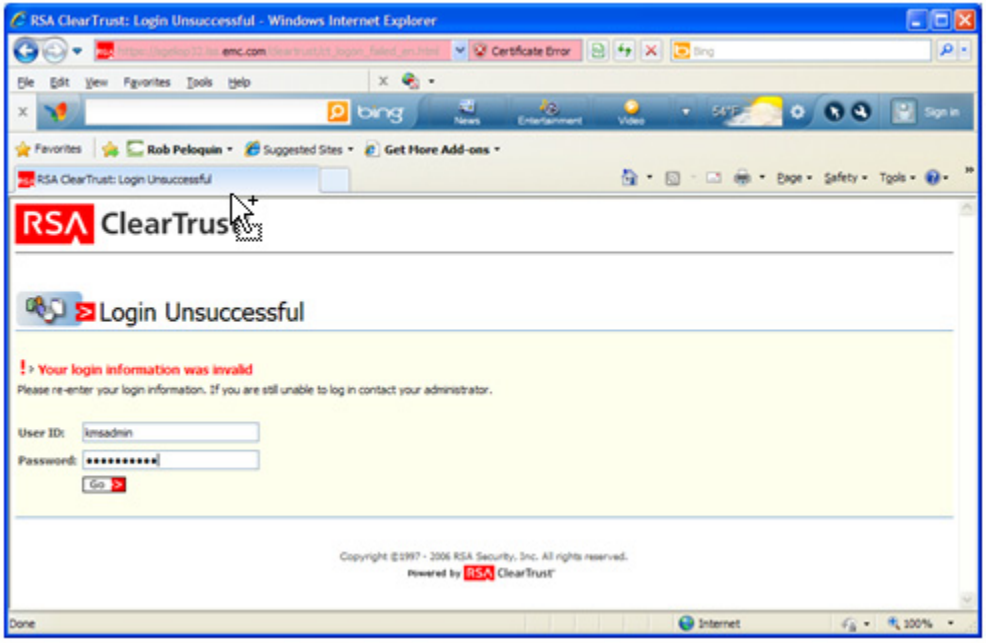
Step 5: Create an Identity for all EG Nodes

Each EG Node at each site must have an Identity created for it on the RKM cluster at that site. This allows the RKMs at each site to identify and authenticate their EG Nodes by their associated KAC certificate.

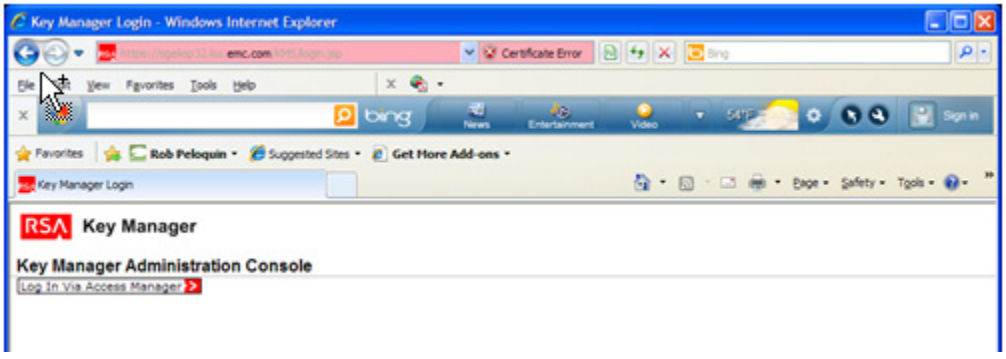
In addition, creating an identity on the RKM server also requires the associated KAC certificate of the encryption node. If it was signed from the RKM server, it can be transferred via SCP or FTP to a workstation on the site. This site should be equipped with a web browser to access the RKM server.

a. At site 1, open a Web browser and connect to the setup page of the primary RKM server using the URL https://rkm_server_network_name/KMS. The rkm_server_network_name is the network name corresponding to the IP address of the primary RKM server at

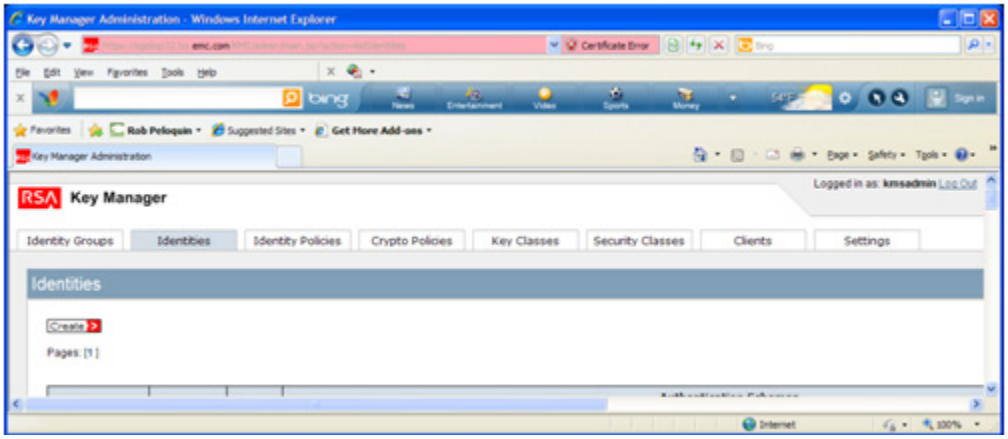
Site 1. In the example shown next, the network name of the primary RKM appliance is sgeliop32.lss.emc.com. The username is kmsadmin.


RSA ClearTrust: Login Unsuccessful - Windows Internet Explorer
RSA ClearTrust
Login Unsuccessful
Your login information was invalid
Please re-enter your login information. If you are still unable to log in contact your administrator.
User ID: kmsadmin
Password:
Go
Copyright ©1997 - 2006 RSA Security, Inc. All rights reserved.
Powered by RSA ClearTrust
Done
Internet
100%

b. Once logged in, select **Log In Via Access Manager** as shown next.


Key Manager Login - Windows Internet Explorer
RSA Key Manager
Key Manager Administration Console
Log In Via Access Manager

c. Click the **Identities** tab, as shown next, and then click **Create**.



d. In the RKM management GUI at each site, create one Identity for every EG Node present.

Remember that site 1 consists of EG Nodes Mace131 and Mace136 and site 2 consists of EG Nodes Mace132 and Mace137. Therefore, at site 1 you will be creating Identities for Mace131 and Mace136 while at site 2 you will be creating Identities for Mace132 and Mace137.

The following steps within the **Identities-Create** page show how to create an Identity:

1. Give each Identity a unique name in the **Name** field.

   – I2051131_mace131 as shown in the example below

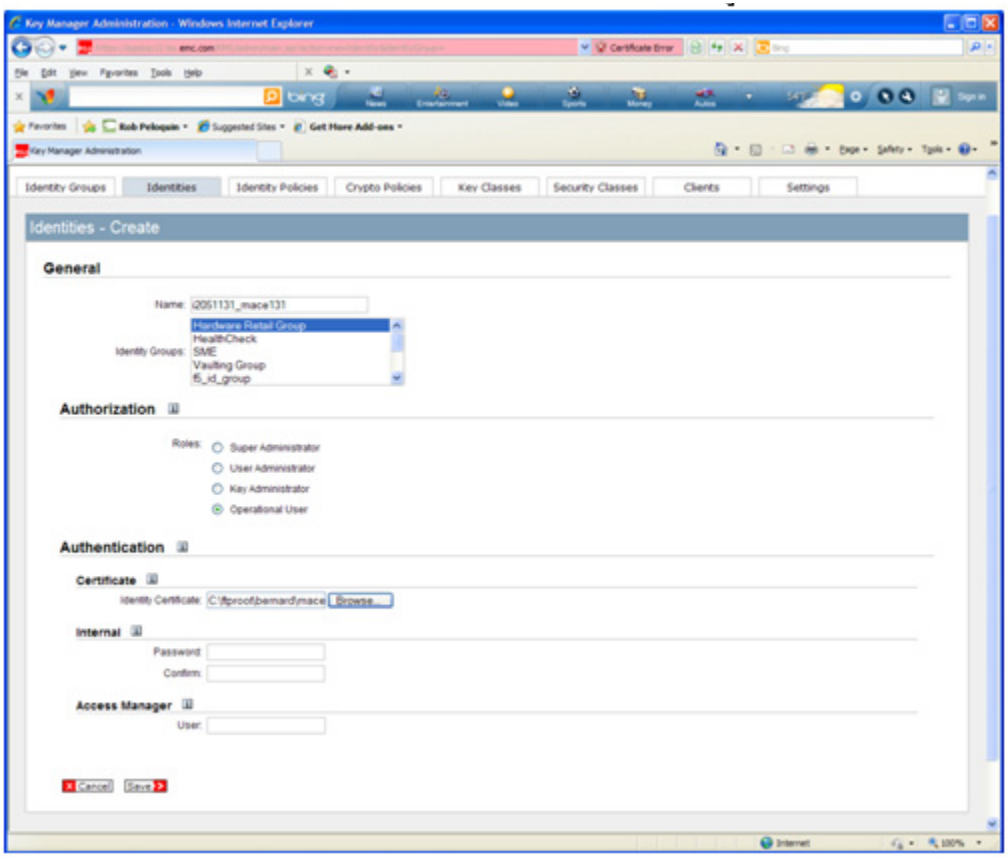
2. Select the **Identity Group** that will be associated with all of the EG Nodes.

   – Hardware Retail Group, as shown in the next example.

3. Ensure the **Authorization Role** is left as the default Operational User.

4. Under **Authentification > Certificate**, use the **Browse** button to browse to the signed KAC certificate that will be associated with the Identity being created.

5. Click **Save**.



Setting up ADX IPLB (IP Load Balance)

The IPLB setup will be performed by professional services as part of the encryption solution installation.

Prior to FOS v6.3.x, key vault HA was provided by having a stand-alone Primary and stand-alone Backup RKM key vault for every EG. The problem with that solution was that there was no synchronization between the two key vaults. Therefore, as of FOS v6.3.x, for high availability, two RKM key vaults per EG are required to be clustered to one another.

Note: Refer to the *EMC Support Matrix* for a complete list of supported RKM KV, FOS, and IPLB software versions.

Without clustered IPLBs in place (which is not supported as an HA solution by EMC), the EG setup would be as follows:

- The IP address of Primary RKM would be registered on the EG leader as the only KV IP address
- The EEs would write/archive DEKs to the Primary RKM only (the one registered for the EG). The Primary RKM would then synchronize the DEKs to its clustered RKM via RKM Cluster Synchronization.
- If it were necessary to retrieve a DEK, all EEs in the EG would do so from the Primary RKM (the one registered on EE).

The problem with the above setup is if the Primary RKM cluster member were to fail or if the Ethernet connectivity from the EG nodes to the Primary RKM were to become unavailable, the following would need to take place:

- Customers or support centers would need to diagnose the situation.
- If this were simply a network connectivity issue, if network connectivity could not be restored, the Primary KV must be de-registered from the EG, and then Secondary Key Vault IP address (the other RKM cluster member) must be registered in its place. Otherwise, the customer can wait until connectivity has been repaired or restored.
- If the Primary RKM cluster member has failed, you must de-register the KV (the primary RKM) from the EG and then register the Secondary Key Vault IP address (access to this newly registered KV will be both Read and Write). Otherwise, the customer can wait until the Primary RKM has been repaired or replaced.

For full details on how to setup the clustered ADX IPLB solution, please refer to the *RSA Key Management (RKM) High Availability, Deployment Guide using the ServerIron ADX 1000* document, which can be located at the Brocade website at http://www.Brocade.com.

The following sections, discussing various IPLB deployment possibilities, were taken directly from the *ADX1000 Deployment Guide:*

- "Topology summary" on page 258
- "Single subnet (single-arm) topology" on page 258
- "Routed subnet topology" on page 259
- "Remote server Subnet topology" on page 261

Topology summary

The ServerIron ADX1000 offers many types of deployment topology options, but the three most commonly deployed topologies are the: Single-Subnet, Routed-Subnet topologies and Remote-Server-Subnet topologies.

Single subnet (single-arm) topology

Configuration overview

In this type of topology, the RKM Servers and the Virtual IP address (VIP) that the Encryption devices will connect to are in the same logical Layer 3 Subnet (in the same IP range). In order for this topology to work and for end users to correctly establish and monitor connections to the RKM servers, Network Address Translation (NAT) must be used so that incoming client sessions are returned back to the ADX.

Figure 35 on page 259 is a basic illustration of a Single Subnet configuration where each ADX1000, RKM Server, and Brocade Encryption device is connected to the same Layer 3 Logical Subnet. The two ADX1000 units are also directly attached to one another to provide a session synchronization.

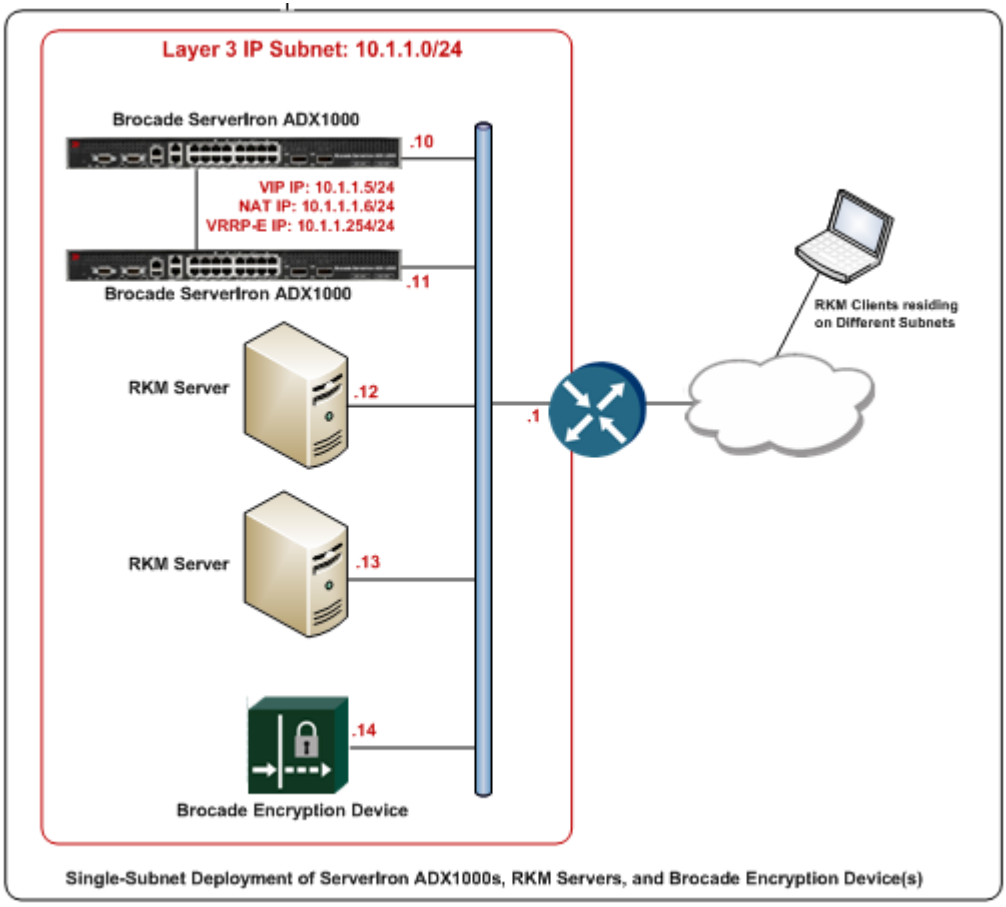


Figure 35 **Single subnet deployment**

Routed subnet topology

Configuration overview

In this type of topology, the RKM Servers and the Virtual IP address (VIP) that the Encryption devices will connect to are two different logical Layer 3 subnets. In order for this topology to work, the RKM servers must have their default-gateways set to the IP address of the ServerIron ADX1000 (using the VRRP IP address). Network Address Translation (NAT) is not a requirement as traffic initiated from the Encryption Device will automatically flow back through the ADX.

In addition, the VIPs will be part of the additional IP subnet that is pointing to the ADX's default gateway.

Figure 36 is a basic illustration of a Routed Subnet (also called a multiple subnet) configuration where the ADX1000 has two different Layer 3 Subnets, one for routing to its default gateway and one for routing to the RKM Servers. The Brocade Encryption Device and RKM clients can be on completely different subnets as long as the Routing on the network is in place to reach the VIP. The two ADX1000 units are also attached to each other to provide session synchronization.

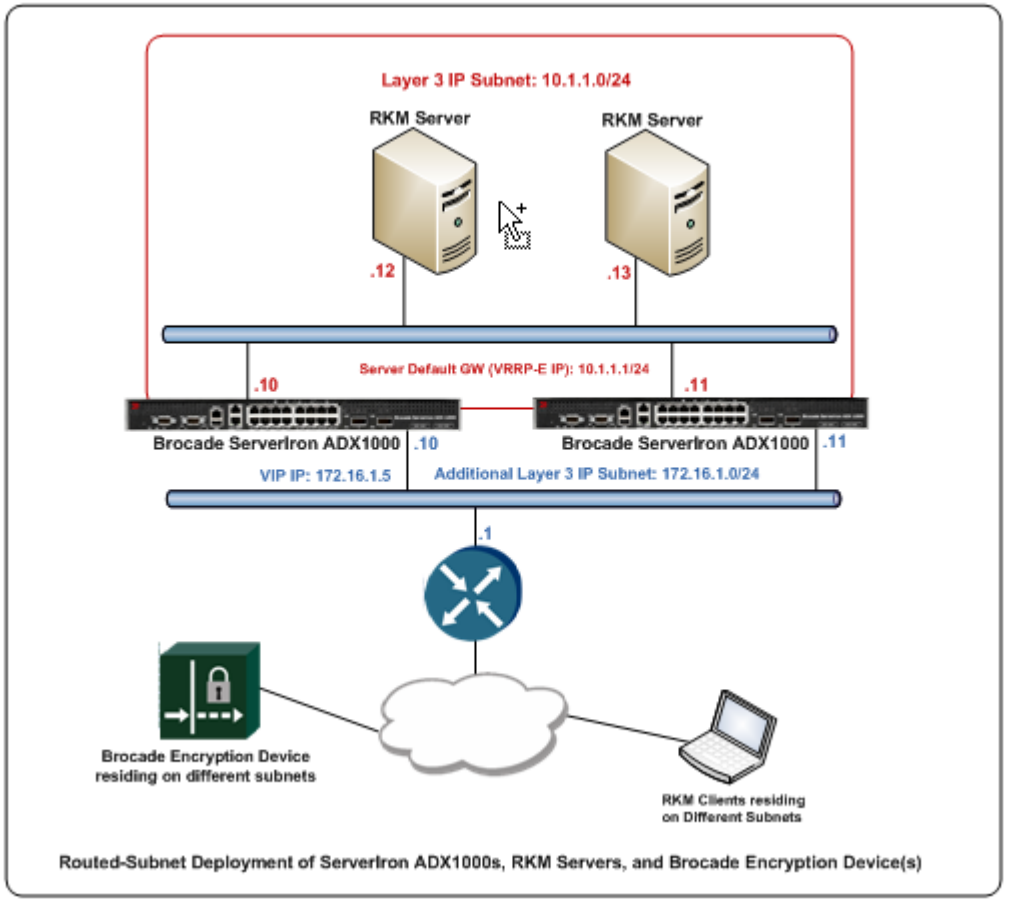


Figure 36 Routed subnet topology

Remote server Subnet topology

Configuration overview

In this type of topology, the RKM Servers and the Virtual IP address (VIP) that the Encryption devices will establish connections to are on two different logical Layer 3 subnets. However, the primary differences between this topology and the Routed Topology is that the RKM servers do not have the ADX set as their default gateway and typically reside on a subnet that is reachable via Layer 3 IP routing. In order for this topology to work, Network Address Translation (NAT) is a requirement as traffic initiated from the Encryption Device will automatically flow back through the ADX.

Figure 37 on page 262 is a basic illustration of a Remote Server Subnet topology configuration where the ADX1000 is on a completely different Layer subnet than the RKM Servers. The Brocade Encryption Device and RKM clients can be on completely different subnets as long as the Routing on the network is in place to reach the VIP. The two ADX1000 units are also attached to each other to provide a failover path.

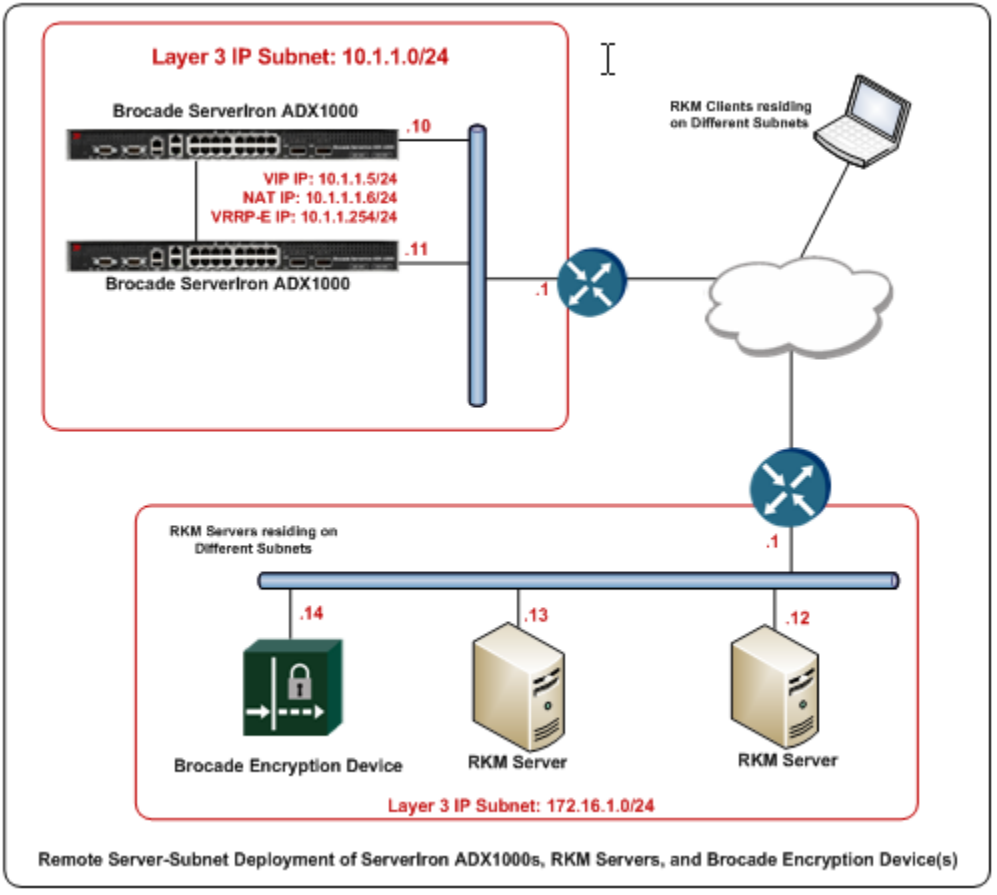


Figure 37 Remote Server Subnet topology

Registering the RKM KV on the Encryption Group leader

Complete the following steps to register the RKM KV on the Encryption Group leader.

Step 1: Import the CA certificate to the EG leader Node

You must now import the CA certificate at each site into the site's EG leader node.

- From Site 1's EG leader node Mace131:

```
i2051131:admin> cryptocfg --import -scp RKMCA.pem 10.32.139.32 root /etc/ssl/cer
ts/trusted_cas.pem
Available Space:4096
Make sure your file size is not greater than 4096.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to proceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.32.139.32's password:
Operation succeeded.
```

- From Site 2's EG leader node Mace132:

```
i2051132:admin> cryptocfg --import -scp RKMCA.pem 10.32.139.32 root /etc/ssl/cer
ts/trusted_cas.pem
Available Space:4096
Make sure your file size is not greater than 4096.
The switch will be unstable or the operation will fail if you exceed this limit.
Do you want to proceed?
ARE YOU SURE  (yes, y, no, n): [no] y
root@10.32.139.32's password:
Operation succeeded.
```

Step 2: Register the RKM KV

You need to register the RKM KVs on each of the two site's EG leader nodes. The IP address used in the following commands will be the IP address configured for the Virtual server created on the clustered ADX 1000 IPLBs; that is, you do not use the IP address of any actual RKM appliance.

In the example below, the clustered IPLBs at Site 1 have utilized a Virtual Server IP address of 10.32.139.200. This is the IP address all EEs at Site 1 will send their DEK read and write request to. The IPLBs will then load balance the requests between the two back-end RKM cluster appliances.

- From Site 1's EG leader node Mace131:

```
i2051131:admin> cryptocfg --reg -keyvault R1_RKMS RKMCA.pem 10.32.139.200 primary
Register key vault status: Operation Succeeded.
```

In the following example, the clustered IPLBs at Site 2 have utilized a Virtual Server IP address of 11.32.139.200. This is the IP address all EEs at Site 2 will send their DEK read and write request to. The IPLBs will then load balance the requests between the two back-end RKM cluster appliances.

- From Site 2's EG leader node Mace132:

```
i2051132:admin> cryptocfg --reg -keyvault R2_RKMS RKMCA.pem 11.32.139.200 primary
Register key vault status: Operation Succeeded.
```

After registering the KV at Site R1, you can check the status of your KV connection to ensure you have a state of Connected. If the state is anything other than Connected, it will be necessary to retrace your configuration setup steps to determine what went wrong:

```
i2051131:admin> cryptocfg --show -groupcfg
Encryption Group Name:R1_131_136
    Failback mode:Auto
    Replication mode:Enabled
    Heartbeat misses:3
    Heartbeat timeout:2
    Key Vault Type:RKM
    System Card:Disabled

Primary Key Vault:
    IP address:10.32.139.200
    Certificate ID:sgeliopvm20
    Certificate label:R1_RKMS
    State:Connected
    Type:RKM

Secondary Key Vault not configured

Additional Key Vault/Cluster Information:
Key Vault/CA Certificate Validity:      Yes
Port for Key Vault Connection:          443
Time of Day on Key Server:              N/A
Server SDK Version:                     N/A

Encryption Node (Key Vault Client) Information:
Node KAC Certificate Validity:          Yes
Time of Day on the Switch:              N/A
Client SDK Version:                     RKM-Client 2.1.1 27-September-2007
Client Username:                        N/A
Client Usergroup:                       N/A
Connection Timeout:                     3 seconds
Response Timeout:                       25 seconds
Connection Idle Timeout:                N/A

Key Vault configuration and connectivity checks successful, ready for key
   operations.


Authentication Quorum Size:0
Authentication Cards not configured


NODE LIST
Total Number of defined nodes:2
Group Leader Node Name:10:00:00:05:1e:c1:75:ac
```

```
Encryption Group state:CLUSTER_STATE_CONVERGED
Crypto Device Config state:In Sync
Encryption Group Config state:In Sync

      Node Name           IP address       Role
10:00:00:05:1e:c1:75:ac  10.246.51.131   GroupLeader  (current node)
    EE Slot:0
    SP state:Operational; Need Valid KEK

10:00:00:05:1e:54:22:75     10.246.51.136         MemberNode
    EE Slot:0
    SP state:Operational; Need Valid KEK
```

After registering the KV at Site R2, you can check the status of your KV connection to ensure you have a state of Connected. If the state is anything other than Connected, it will be necessary to retrace your configuration/setup steps to determine what went wrong:

```
i2051132:admin> cryptocfg --show -groupcfg
Encryption Group Name:R2_132_137
    Failback mode:Auto
    Replication mode:Disabled
    Heartbeat misses:3
    Heartbeat timeout:2
    Key Vault Type:RKM
    System Card:Disabled

Primary Key Vault:
    IP address:11.32.139.200
    Certificate ID:sgeliopvm20
    Certificate label:R2_RKMS
    State:Connected
    Type:RKM

Secondary Key Vault not configured

Additional Key Vault/Cluster Information:
Key Vault/CA Certificate Validity:      Yes
Port for Key Vault Connection:          443
Time of Day on Key Server:              N/A
Server SDK Version:                     N/A

Encryption Node (Key Vault Client) Information:
Node KAC Certificate Validity:          Yes
Time of Day on the Switch:              N/A
Client SDK Version:                     RKM-Client 2.1.1 27-September-2007
Client Username:                        N/A
Client Usergroup:                       N/A
Connection Timeout:                     3 seconds
Response Timeout:                       25 seconds
Connection Idle Timeout:                N/A
```

```
Key Vault configuration and connectivity checks successful, ready for key
   operations.

Authentication Quorum Size:0
Authentication Cards not configured


NODE LIST
Total Number of defined nodes:2
Group Leader Node Name:10:00:00:05:1e:55:88:b7
Encryption Group state:CLUSTER_STATE_CONVERGED
Crypto Device Config state:In Sync
Encryption Group Config state:In Sync

      Node Name IP address   Role
10:00:00:05:1e:55:88:b7    10.246.51.132    GroupLeader  (current node)
    EE Slot:0
    SP state:Operational; Need Valid KEK

10:00:00:05:1e:54:22:44     10.246.51.137    MemberNode
    EE Slot:0
    SP state:Operational; Need Valid KEK
```

Dealing with certificate expiration (KAC, Apache Server, and CA)

Typically all certificate types are created with lifespans. Specifically, the following types of EG solution certificates can expire:

- EG Node KAC certificates
- RKM Apache server certificates

The CA which was used to sign the KAC and Apache certificates can also expire.

It is largely up to the discretion of the professional services installation team working in concert with the customer to determine what the lifespans for the individual certificate types will be. Unfortunately, once a certificate expires, it will no longer be functional which can lead to operational issues within the EG.

As an example, in the following command shows the operand *days*, which has been set to *1825* (1825 days is the equivalent of 5 years):

```
[root@sgeliop32 certs]# openssl x509 -sha1 -req -days 1825 -in ./mace131.csr -CA
 /etc/ssl/certs/trusted_cas.pem -CAkey ./sgeliopvm20-ca-key.pem -CAcreateserial
-out ./mace_131_signed.pem
```

The above example shows that five years after creation of the signed KAC certificate (for EG Node Mace131), it will expire. Once this

happens, all communication between Mace131 and its associated RKM appliances will be lost until a new KAC certificate is created/signed. Unfortunately, when a KAC expires there is no clear means by which to determine that if you are experiencing a connectivity issue between the EG Node and the RKM appliances, that this is the cause of the issue. The connectivity status will simply show a message similar to *Authentification Failed*.

You can use the **openssl** command to determine how much time is left on the validity of any given certificate (KAC, CA, or Apache server). The **openssl** command shown next is used to view the details of the signed mace_131_signed.pem certificate:

```
[root@sgeliop32 certs]# openssl x509 -in ./mace_131_signed.pem -text -noout
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            b8:6a:2b:74:6d:a7:65:c4
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: CN=sgeliopvm20, C=SG, ST=SG, L=SG,
   O=EMC/emailAddress=sgeliopvm20@emc.com
        Validity
            Not Before: Feb 14 21:48:59 2011 GMT
            Not After : Feb 13 21:48:59 2016 GMT
        Subject: CN=kac.000000051ec175ac, C=US, ST=CA, L=San Jose, O=BRCD,
   OU=Technical Support
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
```

Of particular interest in the above example are the *Validity* times. Note that the signed KAC certificate is set to expire on Feb 13 of the year 2016. Once the date Feb 14 of 2016 arrives, communications between this EG Node and its associated RKM cluster is lost.

Also note in the previous output where it reads CN=kac.000000051ec175ac. The last part of this output shows 051ec175ac. This number must match the last 8 digits of the WWN of that particular EG Node. Checking this number ensures that you are looking at the correct KAC certificate.

For example, for Site 1, look at the following partial output of the **cryptocfg –show –groupcfg** command where it lists the WWN of Mace131 (the EG leader node). Note that it matches the KAC output from the above **openssl** command.

```
i2051132:admin>cryptocfg –show –groupcfg

Output truncated….
```

```
NODE LIST
Total Number of defined nodes:2
Group Leader Node Name:10:00:00:05:1e:c1:75:ac
Encryption Group state:CLUSTER_STATE_CONVERGED
Crypto Device Config state:In Sync
Encryption Group Config state:In Sync

      Node Name          IP address       Role
10:00:00:05:1e:c1:75:ac  10.246.51.131   GroupLeader  (current node)
    EE Slot:0
    SP state:Operational; Need Valid KEK

Output truncated….
```

Expired KAC certificates

When an EG Node KAC certificate expires, that EG Node will no longer be able to communicate with its RKM KVs. Dealing with expired KAC certificates is a somewhat straight-forward process. Use the following steps to recover from expired KAC certificates:

- Determine where the original KAC CSR from the EG Node in question is located.
  - If the original KAC CSR cannot be located, use the **cryptocfg –export** command to export a new KAC CSR from the EG Node.
- Sign the CSR.
- From the EG Node, import the newly signed KAC certificate.
- From the EG Node, register the newly signed KAC certificate .
- Go to the RKM cluster KMS GUI.
  - Locate the Identity associated with the EG Node whose KAC certificate has expired.
  - Delete the old KAC certificate associated with the Identity.
  - Associate the newly signed KAC certificate to the Identity.
- Verify whether your connectivity to the RKM KV has been restored by issuing a **cryptocfg –show –groupcfg** command from the EG Node and checking for a Connected status.

Expired Apache server certificates

When an RKM appliance Apache server certificate expires, you will no longer be able to access that RKM appliance's KMS GUI. To determine the location of a given RKM server's Apache server certificate, SSH into the RKM appliance and view the ssl.conf file located in the /etc/httpd/conf.d directory. Within the file, search for

the line which reads *SSLCertificateFile.* This is the RKM appliance Apache server certificate for the appliance. For example:

```
[root@sgeliop32 certs]# cat /etc/httpd/conf.d/ssl.conf

Output truncated…

#   Server Certificate:
#   Point SSLCertificateFile at a PEM encoded certificate.  If
#   the certificate is encrypted, then you will be prompted for a
#   pass phrase.  Note that a kill -HUP will prompt again.  A new
#   certificate can be generated using the genkey(1) command.
SSLCertificateFile /etc/ssl/certs/sgeliop32.lss.emc.com.pem

#   Server Private Key:
#   If the key is not combined with the certificate, use this
#   directive to point at the key file.  Keep in mind that if
#   you've both a RSA and a DSA private key you can configure
#   both in parallel (to also allow the use of DSA ciphers, etc.)
SSLCertificateKeyFile /etc/ssl/private/sgeliop32.lss.emc.com.key

Output truncated…
```

Once you have the location of the Apache server certificate, you can use the same **openssl** command to again determine when it is going to expire. For example:

```
[root@sgeliop32 certs]# openssl x509 -in /etc/ssl/certs/sgeliop32.lss.emc.com.pem
   -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: CN=sgeliopvm20, C=SG, ST=SG, L=SG,
   O=EMC/emailAddress=sgeliopvm20@emc.com
        Validity
            Not Before: Jan  7 10:20:05 2010 GMT
            Not After : Jan  6 10:20:05 2013 GMT
        Subject: C=SG, ST=SG, O=EMC,
   CN=sgeliop32.lss.emc.com/emailAddress=sgeliop32.lss.emc.com@emc.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:c3:7e:c2:92:37:a8:ce:2e:aa:0f:61:23:61:e6:
                    23:bb:08:ee:80:ec:4f:44:a2:eb:f9:a2:7d:84:f1:
```

The above output shows that the Apache server certificate for this RKM appliance will expire on Jan 6 of 2013.

You can deal with expired or expiring Apache server certificates by completing the following steps:

1. Determine where the original Apache server CSR from the RKM appliance is located.

   If the original Apache server CSR cannot be located, use **openssl** to generate a new Apache server CSR. To do this, use the following steps:

   a. Generate a private key for the Apache server. For example:

      ```
      openssl genrsa -out temp/apache_server.key 2048
      ```

   b. Generate a CSR for the Apache server using the above private key. For example:

      ```
      openssl req -new -key temp/apache_server.key -out
         temp/apache_server.csr
      ```

2. Sign the Apache server CSR.
3. Back up, and then edit, the /etc/httpd/conf.d/ssl.conf file, if necessary.

   IMPORTANT

   Where the newly signed Apache server certificate and its private key reside is very important. You will need to edit the /etc/httpd/conf.d/ssl.conf file to make edits to the entries for SSLCertificateFile and SSLCertificateKeyFile if either of their locations change as a result of your work.

4. Restart the Apache web server by using the following command:

   ```
   [root@sgeliop32 certs]# /etc/init.d/httpd restart
   ```

Expired CA certificates

When a CA certificate expires, every EG Node in the EG will no longer be able to communicate with their RKM appliances. Dealing with expired CA certificates is a somewhat cumbersome process.

To determine the location of a given RKM's CA certificate, SSH into the RKM appliance and view the ssl.conf file located in the /etc/httpd/conf.d directory. Within the file, search for the line which reads SSLCACertificateFile. This is the CA utilized by the RKM appliance. For example:

```
[root@sgeliop32 certs]# cat /etc/httpd/conf.d/ssl.conf

Partial output shown….
```

```
#   Certificate Authority (CA):
#   Set the CA certificate verification path where to find CA
#   certificates for client authentication or alternatively one
#   huge file containing all of them (file must be PEM encoded)
SSLCACertificateFile /etc/ssl/certs/trusted_cas.pem

#   Client Authentication (Type):
#   Client certificate verification type and depth.  Types are

Output truncated……
```

The same type of **openssl** command can be used to determine when your CA certificate will expire. For example:

```
[root@sgeliop32 certs]# openssl x509 -in ./trusted_cas.pem -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            a3:fa:0c:f8:83:16:ad:2a
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: CN=sgeliopvm20, C=SG, ST=SG, L=SG,
   O=EMC/emailAddress=sgeliopvm20@emc.com
        Validity
            Not Before: Jan  7 10:17:49 2010 GMT
            Not After : Jan  6 10:17:49 2015 GMT
        Subject: CN=sgeliopvm20, C=SG, ST=SG, L=SG,
   O=EMC/emailAddress=sgeliopvm20@emc.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (2048 bit)
                Modulus (2048 bit):
                    00:c9:57:21:00:04:5e:e2:f2:de:ca:32:71:30:ee:
                    70:78:ca:1f:17:c6:26:ba:aa:5b:22:55:0f:2d:f7:
                    cd:6e:a1:3e:96:c1:2f:9e:45:94:76:2d:d7:14:99:
```

The above output shows that the CA certificate for this RKM appliance will expire on Jan 6 of 2015.

Use the following steps to recover from a CA which has expired or which is about to expire:

- Repeat the process outlined in "Expired KAC certificates" on page 268 for every EG Node which had its KAC CSR signed by the expired CA.
- Repeat the process outlined in"Expired Apache server certificates" on page 268 for every RKM appliance which had its Apache server CSR signed by the expired CA.

Generating and backing up the EG Master key

At this point, communication between all EG members at each site and their respective RKM clusters has been established. The final step in preparation before creating your Crypto Target Containers (CTCs) and respective LUNs to be encrypted is to create and distribute the Master key.

The Master key is used to encrypt and decrypt DEKs when storing DEKs to RKM key vaults. There is one master key per EG; therefore all EEs within a EG use the same Master key to encrypt and decrypt the DEKs.

For this solution, SRDF will be used to replicate encrypted LUN data between Site 1 and Site 2. The two RKM clusters at each site will maintain the same database of DEKs for the encrypted LUNs. The two sites are synchronized by using Oracle streams, which are built into the RKM appliance code.

Because a Site 2 EG member may have to decrypt a DEK originally archived to Site 1, the Site 2 EG members must use the same Master key as the Site 1 EG members. Otherwise, upon retrieving an archived DEK, they would not be able to decrypt it.

To make this work, the following steps have to be performed. Each of these steps is discussed in more detail in this section:

- The Master Key will be generated at Site 1 by the Site 1 EG leader node.
- The Master Key at Site 1 will be backed-up up by the Site 1 EG leader node to the Site 1 RKM cluster.
- The Site 1 RKM cluster will automatically replicate this Master Key over to the Site 2 RKM cluster.
- The Site 2 EG leader node will do a **cryptocfg -recovermasterkey** operation to import and utilize the Master key which was generated by the Site 1 EG leader.

Step 1: Generate the Site 1 Master key

Generate the Master key from the Site 1 EG leader node.

From the Site 1 EG leader:

```
i2051131:admin> cryptocfg --genmasterkey
Master key generated.  The master key must be exported before further operations
   are performed.
```

Note: All DEKs archived to the RKM key vaults are encrypted with a Master Key. This Master Key, once generated by the EG leader, must be backed up before the EG is allowed to perform data encryption operations.

Step 2: Back up the Site 1 Master key

Back up the Master key from the Site 1 EG leader node.

From the Site 1 EG leader:

```
i2051131:admin> cryptocfg --exportmasterkey
Enter passphrase: <INSERT PASSPHRASE HERE>
Confirm passphrase: <INSERT SAME PASSPHRASE HERE>

Master key exported.  Key ID: 27:48:5d:8e:af:7c:91:5f:09:0c:bc:84:97:8b:e4:24
```

IMPORTANT

A passphrase is a sequence of words or other text, similar to a password, but generally longer for added security. It is imperative to remember the passphrase as it will be required whenever a Master key is being restored to an EG.

IMPORTANT

Ensure that you keep a backup of the Master Key, because without it you cannot decrypt any DEKs retrieved from RKM key vaults and existing encrypted data can no longer be decrypted.

IMPORTANT

Be careful when choosing the method to store the Master Key. You can back up or restore the Master Key to the key vault, to a file, or to a set of smart cards. Using the smart card set is the most secure approach. There is no CLI option to back up the Master Key to a set of smart cards. This is only available through the GUI.

- Before generating and archiving the Site 1 Master key:

```
i2051131:admin> cryptocfg --show -groupcfg
Encryption Group Name:R1_131_136
    Failback mode:Auto
    Replication mode:Enabled
    Heartbeat misses:3
    Heartbeat timeout:2
    Key Vault Type:RKM
    System Card:Disabled
```

```
Primary Key Vault:
    IP address:10.32.139.200
    Certificate ID:sgeliopvm20
    Certificate label:R1_RKMS
    State:Connected
    Type:RKM

Secondary Key Vault not configured

Additional Key Vault/Cluster Information:
Key Vault/CA Certificate Validity:      Yes
Port for Key Vault Connection:          443
Time of Day on Key Server:              N/A
Server SDK Version:                     N/A

Encryption Node (Key Vault Client) Information:
Node KAC Certificate Validity:          Yes
Time of Day on the Switch:              N/A
Client SDK Version:                     RKM-Client 2.1.1 27-September-2007
Client Username:                        N/A
Client Usergroup:                       N/A
Connection Timeout:                     3 seconds
Response Timeout:                       25 seconds
Connection Idle Timeout:                N/A

Key Vault configuration and connectivity checks successful, ready for key
   operations.


Authentication Quorum Size:0
Authentication Cards not configured


NODE LIST
Total Number of defined nodes:2
Group Leader Node Name:10:00:00:05:1e:c1:75:ac
Encryption Group state:CLUSTER_STATE_CONVERGED
Crypto Device Config state:In Sync
Encryption Group Config state:In Sync

      Node Name IP address   Role
10:00:00:05:1e:c1:75:ac        10.246.51.131      GroupLeader  (current node)
    EE Slot:0
    SP state:Operational; Need Valid KEK

10:00:00:05:1e:54:22:75         10.246.51.136         MemberNode
    EE Slot:0
    SP state:Operational; Need Valid KEK
```

Note: The above output shows both Nodes comprising the EG are in an SP state of Operational; Need Valid KEK. KEK is Key Encryption Key and indicates that there is no Master Key present.

- After generating and archiving the Site 1 Master Key:

```
i2051131:admin> cryptocfg -show -groupcfg
Encryption Group Name:R1_131_136
    Failback mode:Auto
    Replication mode:Enabled
    Heartbeat misses:3
    Heartbeat timeout:2
    Key Vault Type:RKM
    System Card:Disabled

Primary Key Vault:
    IP address:10.32.139.200
    Certificate ID:sgeliopvm20
    Certificate label:R1_RKMS
    State:Connected
    Type:RKM

Secondary Key Vault not configured

Additional Key Vault/Cluster Information:
Key Vault/CA Certificate Validity:      Yes
Port for Key Vault Connection:          443
Time of Day on Key Server:              N/A
Server SDK Version:                     N/A

Encryption Node (Key Vault Client) Information:
Node KAC Certificate Validity:          Yes
Time of Day on the Switch:              N/A
Client SDK Version:                     RKM-Client 2.1.1 27-September-2007
Client Username:                        N/A
Client Usergroup:                       N/A
Connection Timeout:                     3 seconds
Response Timeout:                       25 seconds
Connection Idle Timeout:                N/A

Key Vault configuration and connectivity checks successful, ready for key
   operations.


Authentication Quorum Size:0
Authentication Cards not configured

NODE LIST
Total Number of defined nodes:2
Group Leader Node Name:10:00:00:05:1e:c1:75:ac
Encryption Group state:CLUSTER_STATE_CONVERGED
```

```
Crypto Device Config state:In Sync
Encryption Group Config state:In Sync

      Node Name IP address   Role
10:00:00:05:1e:c1:75:ac     10.246.51.131        GroupLeader  (current node)
    EE Slot:0
    SP state:Online

10:00:00:05:1e:54:22:75          10.246.51.136            MemberNode
    EE Slot:0
    SP state:Online
```

Step 3: Recover the Master key for the Site 2 EG

Once the Master key is archived to the Site 1 RKM cluster, the Site 1 RKM cluster automatically replicated the Master Key over to the Site 2 RKM cluster.

Recover the Master key (generated at Site 1) from the Site 2 EG leader node.

From the Site 2 EG leader:

```
i2051132:admin > cryptocfg --recovermasterkey currentMK -keyID
27:48:5d:8e:af:7c:91:5f:09:0c:bc:84:97:8b:e4:24
Enter the passphrase: <INSERT PASSPHRASE USED TO ARCHIVE THE SITE 1 MASTER KEY>
Recover master key status: Operation succeeded.
```

Ensuring that both fabrics are set for defzone--noaccess

Before committing any encryption configurations in a fabric, the default zoning for that fabric must be set to **No Access**. The No Access setting ensures that no two devices on the fabric can communicate with one another without going through a regular zone or a redirection zone.

Use the following command on all fabrics to check the default zoning setting. Commonly, it will be set to **All Access**:

```
i2051131:admin> defzone --show
Default Zone Access Mode
committed - All Access
transaction - No Transaction
```

If the default zoning is set to **All Access**, change it to **No Access** by using the following command from the primary FCS switch in that fabric, as shown next:

```
i2051131:admin> defzone --noaccess
i2051131:admin> cfgfsave
The change will be applied within the entire fabric.
```

After setting the defzone to **No Access**:

```
i2051131:admin> defzone --show
Default Zone Access Mode
committed - No Access
transaction - No Transaction
```

Enabling remote replication mode

To enable the remote replication features of the Brocade encryption solution, the command **cryptocfg --set -replication enable** must be issued.

This mode can be enabled only when the configured key vault is RKM. The remote replication features are supported in Fabric OS version 6.4 and later. Remote replication is disallowed under the following conditions:

- One of the nodes in an EG is running an earlier Fabric OS version
- A node is downgraded to an earlier Fabric OS version

◆ To enable replication mode on Site 1:

```
i2051131:admin> cryptocfg --set -replication enable
Set replication mode status: Operation Succeeded.
```

◆ To enable replication mode on Site 2:

```
i2051132:admin> cryptocfg --set -replication enable
Set replication mode status: Operation Succeeded.
```

◆ After setting replication mode to enabled:

```
i2051131:admin> cryptocfg --show -groupcfg
Encryption Group Name:  R1_131_136
    Failback mode:      Auto
    Replication mode:   Enabled
    Heartbeat misses:   3
    Heartbeat timeout:  2
    Key Vault Type:     RKM
    System Card:        Disabled

Output truncated….
```

Creating the CTCs at each site

The CTCs which will be part of the encrypted traffic flows must now be added to the EEs at each site. For the purposes of this reference architecture, only the CTCs which are a part of each Site's Fabric A will be displayed below. When setting up this solution for a customer, care must be taken to ensure that all CTCs from all fabrics participating in the SRDF solution are added to the appropriate EEs. Put another way, care must be taken to ensure that every path possible to an encrypted LUN must be owned by an EE.

IMPORTANT

Before configuring the CTCs for each site, ensure that standard zoning has been put in place for initiator and storage ports being added as CTCs.

For this SRDF encryption solution, two CTCs (Symmetrix storage ports) will be added to each Site's Fabric A configuration. One of the two CTCs at each site will be owned by one of the two HA cluster members at each site thereby making each HA cluster member active.

From the Site 1 EG leader node:

```
i2051131:admin> cryptocfg --create -container disk Symm_10G0_R1
   10:00:00:05:1e:c1:75:ac 50:00:09:72:08:2f:45:a4 50:00:09:72:08:2f:44:00
                           Slot     Local/
      EE Node WWN         Number    Remote
10:00:00:05:1e:c1:75:ac      0      Local
Operation succeeded.

i2051131:admin> cryptocfg --add -initiator Symm_10G0_R1 10:00:00:00:c9:8f:58:58
   20:00:00:00:c9:8f:58:58
Operation succeeded.
```

The initiator is added at the CTC level to allow discovery of the LUNs behind the storage port to which the added initiator has access.

```
i2051131:admin> cryptocfg --create -container disk Symm_10G1_R1
   10:00:00:05:1e:54:22:75 50:00:09:72:08:2f:45:a5 50:00:09:72:08:2f:44:00
                           Slot     Local/
      EE Node WWN         Number    Remote
10:00:00:05:1e:54:22:75 0      Local
Operation succeeded.

i2051131:admin> cryptocfg --add -initiator Symm_10G1_R1 10:00:00:00:c9:8f:58:58
   20:00:00:00:c9:8f:58:58
Operation succeeded.
```

Once the commit operation, as shown next is performed, all I/O to all LUNs via the configured storage port (CTCs) will be stopped. This is because the commit operation results in Frame Redirection occurring throughout the fabrics. Therefore, any I/O destined to the configured storage ports will now be going through the EEs and because the EEs have yet to be configured with actual LUNs, all I/O will be halted.

```
i2051131:admin> cryptocfg --commit
Operation succeeded.
```

A maximum of 25 transactions per commit **cryptocfg --commit** can be executed.

```
i2051131:admin> cryptocfg --show -container -all -cfg
Encryption group name:  R1_131_136
Number of Container(s): 2

Container name:         Symm_10G0_R1
Type:                   disk
EE node:                10:00:00:05:1e:c1:75:ac
EE slot:                0
Target:                 50:00:09:72:08:2f:45:a4 50:00:09:72:08:2f:44:00
VT:                     20:02:00:05:1e:54:22:40 20:03:00:05:1e:54:22:40
Number of host(s):      1
Configuration status:   committed
Host:                   10:00:00:00:c9:8f:58:58 20:00:00:00:c9:8f:58:58
VI:                     20:04:00:05:1e:54:22:40 20:05:00:05:1e:54:22:40
Number of LUN(s):       0

Container name:         Symm_10G1_R1
Type:                   disk
EE node:                10:00:00:05:1e:54:22:75
EE slot:                0
Target:                 50:00:09:72:08:2f:45:a5 50:00:09:72:08:2f:44:00
VT:                     20:00:00:05:1e:54:22:40 20:01:00:05:1e:54:22:40
Number of host(s):      1
Configuration status:   committed
Host:                   10:00:00:00:c9:8f:58:58 20:00:00:00:c9:8f:58:58
VI:                     20:06:00:05:1e:54:22:40 20:07:00:05:1e:54:22:40
Number of LUN(s):       0
Operation succeeded.
```

From the Site 2 EG leader node, add the two CTCs associated with that Site's Fabric A. The following output shows the results of the two CTC creations:

```
i2051132:admin> cryptocfg --show -container -all -cfg
Encryption group name:  R2_132_137
Number of Container(s): 2

Container name:         Symm_9F0_R2
Type:                   disk
```

```
EE node:                10:00:00:05:1e:55:88:b7
EE slot:                0
Target:                 50:00:09:72:08:2f:35:60 50:00:09:72:08:2f:34:00
VT:                     20:06:00:05:1e:55:88:ba 20:07:00:05:1e:55:88:ba
Number of host(s):      1
Configuration status:   committed
Host:                   10:00:00:00:c9:8f:58:59 20:00:00:00:c9:8f:58:59
VI:                     20:04:00:05:1e:55:88:ba 20:05:00:05:1e:55:88:ba
Number of LUN(s):       0

Container name:         Symm_9F1_R2
Type:                   disk
EE node:                10:00:00:05:1e:54:22:44
EE slot:                0
Target:                 50:00:09:72:08:2f:35:61 50:00:09:72:08:2f:34:00
VT:                     20:0a:00:05:1e:55:88:ba 20:0b:00:05:1e:55:88:ba
Number of host(s):      1
Configuration status:   committed
Host:                   10:00:00:00:c9:8f:58:59 20:00:00:00:c9:8f:58:59
VI:                     20:0c:00:05:1e:55:88:ba 20:0d:00:05:1e:55:88:ba
Number of LUN(s):       0
Operation succeeded.
```

Adding the source SRDF LUNs to each CTC at Site 1

There are two possible LUN configuration scenarios to consider in encryption SRDF deployments:

- Creating new source LUNs, which can later be replicated.
- Migrating data from existing encrypted or cleartext source LUNs to LUNs which can be replicated.

This solution will create new source LUNs and replicate them to a remote site. However, for each of the above two scenarios, the following rules and notes apply:

- SRDF R1 and R2 LUNs must be of the same size.
- When changing encryption policies for the source LUN, the same policies must be applied to the target LUN.
- Once the LUN is added to the container using the **-newLUN** option, it must not be resized.
- Auto/Key expire rekey is not allowed for SRDF LUNs. Therefore, the **-newLUN** option is not compatible with **-enable_rekey** option.

Now that the CTCs have been created and their configurations committed, you can use the **cryptocfg -discoverLUN** command for

LUN discovery. This discoverLUN command is used on the CTC level and will discover and display all LUNs behind the storage port, for which Initiators that are a part of the CTC have access.

The following output shows that only LUN #0x1 is accessible through the indicated CTC by the Host with PWWN 10:00:00:00:c9:8f:58:58. This command can only be run from the EE which owns CTC Symm_10G0_R1:

```
i2051131:admin> cryptocfg --discoverLUN Symm_10G0_R1
Container name:          Symm_10G0_R1
Number of LUN(s):                1
Host:                    10:00:00:00:c9:8f:58:58
LUN number:              0x1
LUN serial number:       60000970000192603025533030343837
LUN connectivity state: Connected
Key ID state:            Key ID not available

Operation succeeded.
```

The following output shows the same LUN as being accessible from the other CTC Symm_10G1_R1. Note that the LUN number is the same and, more importantly, that the LUN serial number is identical to that of the previous EEs discoverLUN output. This command can only be run from the EE, which owns CTC Symm_10G1_R1:

```
i2051136:admin> cryptocfg --discoverLUN Symm_10G1_R1
Container name:          Symm_10G1_R1
Number of LUN(s):                1
Host:                    10:00:00:00:c9:8f:58:58
LUN number:              0x1
LUN serial number:       60000970000192603025533030343837
LUN connectivity state: Connected
Key ID state:            Key ID not available

Operation succeeded.
```

Note: At this point, we are creating new source LUNs which will be replicated by SRDF to the remote site. If the LUNs had existing customer data on them, they would need to be migrated to larger LUNs to make room for encryption metadata. The process would include creating new/additional LUNs which were larger by three blocks, adding those LUNs with the **-newLUN** option to the same CTCs, and then using EMC PowerPath Migration Enabler (PPME) to copy the data from the existing LUN to the larger LUN. A detailed explanation of this process is included in the *Fabric OS Encryption Administrator's Guide Supporting RSA Key Manager (RKM) Environments, Supporting Fabric OS v6.4.0,* which can be located under the Documents section of MyBrocade located at https://login.brocade.com.

- From the Site 1 EG leader:

IMPORTANT

The cryptocfg -add -LUN command must be completed once for every path or container that has access to the source LUN. Forgetting or missing a path could result in data corruption on the LUN.

Note: Alternatively, the Connectrix Manager Data Center Edition (CMDCE v10.4.x or later GUI can be utilized to add or modify parameters on all paths to a given LUN at the same time.

```
i2051131:admin> cryptocfg --add -LUN Symm_10G0_R1 0x1 10:00:00:00:c9:8f:58:58
   20:00:00:00:c9:8f:58:58 -newLUN -lunstate cleartext -encrypt
Adding LUN with '-newLUN' option will render last 3 blocks on the LUN unusable
ARE YOU SURE  (yes, y, no, n): [no] y
Operation succeeded.
```

IMPORTANT

The above command assumes there is no existing or valid user data on the LUN. Therefore, this will have the effect of destroying any existing user data on the LUN. This is because the default option is disable_encexistingdata, resulting in any existing data on the LUN being lost. If the LUN had existing data on it, migration to the larger LUNwould be necessary to accommodate the Brocade secondary metadata and maintain the integrity of the existing user data. Detailed explanation for the migration process can be found in the *Fabric OS Encryption Administrator's Guide Supporting RSA Key Manager (RKM) Environment, Supporting Fabric OS v6.4.0* document at https://login.brocade.com.

```
i2051131:admin> cryptocfg --add -LUN Symm_10G1_R1 0x1 10:00:00:00:c9:8f:58:58
   20:00:00:00:c9:8f:58:58 -newLUN -lunstate cleartext -encrypt
Adding LUN with '-newLUN' option will render last 3 blocks on the LUN unusable
ARE YOU SURE  (yes, y, no, n): [no] y
Operation succeeded.

i2051131:admin> cryptocfg --commit
Operation succeeded.
```

- After adding the LUN as shown from Mace131:

```
i2051131:admin> cryptocfg --show -container -all -stat
Encryption group name:  R1_131_136
Number of Container(s): 1
Container name:         Symm_10G0_R1
```

```
Type:                    disk
EE node:                 10:00:00:05:1e:c1:75:ac
EE slot:                 0
EE hosting container:    current
Target:                  50:00:09:72:08:2f:45:a4 50:00:09:72:08:2f:44:00
Target PID:              010500
VT:                      20:02:00:05:1e:54:22:40 20:03:00:05:1e:54:22:40
VT PID:                  012001
Number of host(s):       1
Number of rekey session(s):     0
Host:                    10:00:00:00:c9:8f:58:58 20:00:00:00:c9:8f:58:58
Host PID:                011000
VI:                      20:04:00:05:1e:54:22:40 20:05:00:05:1e:54:22:40
VI PID:                  012401
Number of LUN(s):        1
LUN number:              0x1
LUN type:                disk
LUN serial number:       60000970000192603025533030343837
Encryption mode:         encrypt
Encryption format:       native
Encrypt existing data:   disabled
Rekey:                   disabled
Internal EE LUN state:   Encryption enabled
Encryption algorithm:    AES256-XTS
Key ID state:            Read write
New LUN:                 Yes
Replication LUN type:    Primary
Key ID:                  84:92:ad:e3:51:df:c9:af:94:8f:88:80:f5:89:75:a9
Key creation time:       Mon Feb 21 19:01:33 2011
Operation succeeded.
i2051131:admin>
```

- After adding the LUN as shown from Mace136:

```
i2051136:admin> cryptocfg --show -container -all -stat
Encryption group name:  R1_131_136
Number of Container(s): 1

Container name:          Symm_10G1_R1
Type:                    disk
EE node:                 10:00:00:05:1e:54:22:75
EE slot:                 0
EE hosting container:    current
Target:                  50:00:09:72:08:2f:45:a5 50:00:09:72:08:2f:44:00
Target PID:              030100
VT:                      20:00:00:05:1e:54:22:40 20:01:00:05:1e:54:22:40
VT PID:                  032101
Number of host(s):       1
Number of rekey session(s):     0
Host:                    10:00:00:00:c9:8f:58:58 20:00:00:00:c9:8f:58:58
Host PID:                011000
VI:                      20:06:00:05:1e:54:22:40 20:07:00:05:1e:54:22:40
VI PID:                  032401
```

```
Number of LUN(s):       1
LUN number:             0x1
LUN type:               disk
LUN serial number:      600009700001926030255330303438 37
Encryption mode:        encrypt
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Encryption enabled
Encryption algorithm:   AES256-XTS
Key ID state:           Read write
New LUN:                Yes
Replication LUN type:   Primary
Key ID:                 84:92:ad:e3:51:df:c9:af:94:8f:88:80:f5:89:75:a9
Key creation time:      Mon Feb 21 19:01:33 2011
Operation succeeded.
```

Adding the source SRDF LUNs to each CTC at Site 2

This section describes the proper procedure for establishing the local/remote LUN pair in an SRDF environment.

Note: The remote/target LUNs must be added to their CryptoTarget Containers (CTCs) only after the local site LUNs' encryption setup has been completed.

Establish the local to remote LUN replication/synchronization and wait for the pair to become fully synchronized. You can verify whether the SRDF pair is in a synchronized state using the EMC Solution Enabler.

Note: During the initial SRDF replication (or while replicating or synchronizing after a rekey of the source LUN), the remote/R2 LUNs must not be exposed for access to the remote site hosts.

Although the SRDF behavior may make the remote/R2 LUN read-only or not-ready, it is mandated that the target ports be physically taken offline. Once synchronized, if remote access to the target LUN becomes necessary, the process of bringing the remote target ports online will ensure that the correct Data Encryption Key (DEK) is injected into every Encryption Engine (EE) with a path to the remote LUN.

- From the Site 2 EG leader:

```
i2051132:admin> cryptocfg --add -LUN Symm_9F0_R2 0x1 10:00:00:00:c9:8f:58:59
   20:00:00:00:c9:8f:58:59 -lunstate encrypted -encrypt -newLUN
Operation succeeded.
```

Note: The syntax above is slightly different on the remote site. The lun state is now encrypted as opposed to cleartext as it was on the R1 site. This is because SRDF has replicated the encrypted data from site R1 to site R2. Therefore, when adding the Site 2 LUN, it must be added as an encrypted LUN.

```
i2051132:admin> cryptocfg --add -LUN Symm_9F1_R2 0x1 10:00:00:00:c9:8f:58:59
   20:00:00:00:c9:8f:58:59 -lunstate encrypted -encrypt -newLUN
Operation succeeded.

i2051132:admin> cryptocfg --commit
Operation succeeded.
```

- After adding the LUN as shown from Mace132:

```
i2051132:admin> cryptocfg --show -container -all -stat
Encryption group name:  R2_132_137
Number of Container(s): 1

Container name:         Symm_9F0_R2
Type:                   disk
EE node:                10:00:00:05:1e:55:88:b7
EE slot:                0
EE hosting container:   current
Target:                 50:00:09:72:08:2f:35:60 50:00:09:72:08:2f:34:00
Target PID:             020200
VT:                     20:06:00:05:1e:55:88:ba 20:07:00:05:1e:55:88:ba
VT PID:                 022c01
Number of host(s):      1
Number of rekey session(s):     0
Host:                   10:00:00:00:c9:8f:58:59 20:00:00:00:c9:8f:58:59
Host PID:               020100
VI:                     20:04:00:05:1e:55:88:ba 20:05:00:05:1e:55:88:ba
VI PID:                 022401
Number of LUN(s):       1
LUN number:             0x1
LUN type:               disk
LUN serial number:      60000970000192603021533030363830
Encryption mode:        encrypt
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Encryption enabled
Encryption algorithm:   AES256-XTS
Key ID state:           Read write
New LUN:                Yes
Replication LUN type:   Mirror
Key ID:                 84:92:ad:e3:51:df:c9:af:94:8f:88:80:f5:89:75:a9
Key creation time:      Mon Feb 21 19:01:33 2011
Operation succeeded.
```

- After adding the LUN as shown from Mace137:

```
i2051137:admin> cryptocfg --show -container -all -stat
Encryption group name:  R2_132_137
Number of Container(s): 1

Container name:         Symm_9F1_R2
Type:                   disk
EE node:                10:00:00:05:1e:54:22:44
EE slot:                0
EE hosting container:   current
Target:                 50:00:09:72:08:2f:35:61 50:00:09:72:08:2f:34:00
Target PID:             040100
VT:                     20:0a:00:05:1e:55:88:ba 20:0b:00:05:1e:55:88:ba
VT PID:                 042101
Number of host(s):      1
Number of rekey session(s):     0
Host:                   10:00:00:00:c9:8f:58:59 20:00:00:00:c9:8f:58:59
Host PID:               020100
VI:                     20:0c:00:05:1e:55:88:ba 20:0d:00:05:1e:55:88:ba
VI PID:                 042501
Number of LUN(s):       1
LUN number:             0x1
LUN type:               disk
LUN serial number:      60000970000192603021533030363830
Encryption mode:        encrypt
Encryption format:      native
Encrypt existing data:  disabled
Rekey:                  disabled
Internal EE LUN state:  Encryption enabled
Encryption algorithm:   AES256-XTS
Key ID state:           Read write
New LUN:                Yes
Replication LUN type:   Mirror
Key ID:                 84:92:ad:e3:51:df:c9:af:94:8f:88:80:f5:89:75:a9
Key creation time:      Mon Feb 21 19:01:33 2011
Operation succeeded.
```

Take all remote target ports associated with CTCs through which the remote LUNs are accessible offline.

You have now completed the process of setting up an SRDF encryption environment.

A very useful command when configuring or troubleshooting encryption is the **-show -egstatus** command. This has both the **-cfg** and **-stat** options. Each output is shown next.

```
i2051131:admin> cryptocfg --show -egstatus -cfg | more

Encryption Group Details:
---------------------------------

Encryption Group Information
            EG Name:                    R1_131_136
            EG state:                   CLUSTER_STATE_CONVERGED
            Failback mode:              Auto
            Replication mode:           Enabled
            Heartbeat misses:           3
            Heartbeat timeout:          2
            Number of defined nodes:    2
            Group Leader Node Name:     10:00:00:05:1e:c1:75:ac

        Node Name:                      10:00:00:05:1e:c1:75:ac  (current node)
            State:                      DEF_NODE_STATE_DISCOVERED
            Role:                       GroupLeader
            IP Address:                 10.246.51.131
            Certificate:                10.246.51.131_my_cp_cert.pem
            Current Master Key State:   Saved
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
            Alternate Master Key State: Saved
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

     EE Slot:                   0
         SP state:              Online
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
            HA Cluster Membership:      R1_131_136_cluster

        Node Name:                      10:00:00:05:1e:54:22:75
            State:                      DEF_NODE_STATE_DISCOVERED
            Role:                       MemberNode
            IP Address:                 10.246.51.136
            Certificate:                10.246.51.136_my_cp_cert.pem
            Current Master Key State:   Saved
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
            Alternate Master Key State: Not configured
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

     EE Slot:                   0
         SP state:              Online
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
```

```
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
            HA Cluster Membership:      R1_131_136_cluster

Key Vault Details:
----------------------------------

  Primary Key Vault Information:
    Key Vault State:    configured
    IP address:         10.32.139.200
    Certificate ID:     sgeliopvm20
    Certificate label:  R1_RKMS
    State:              Connected
    Type:               RKM

  Secondary Key Vault Information:
    Key Vault State:    not configured

Additional Key Vault/Cluster Information:
        Key Vault/CA Certificate Validity:      Yes
        Port for Key Vault Connection:          443
        Time of Day on Key Server:              N/A
        Server SDK Version:                     N/A

Encryption Node (Key Vault Client) Information:
        Node KAC Certificate Validity:          Yes
        Time of Day on the Switch:              N/A
       Client SDK Version:                   RKM-Client 2.1.1 27-September-2007
        Client Username:                        N/A
        Client Usergroup:                       N/A
        Connection Timeout:                     3 seconds
        Response Timeout:                       25 seconds
        Connection Idle Timeout:                N/A

Key Vault configuration and connectivity checks successful, ready for key
   operations.

HA Cluster Details:
----------------------------------

HA Cluster State:       Configured
Number of HA Clusters:  1

HA cluster name: R1_131_136_cluster - 2 EE entries
Status:          Committed
HAC State:       Converged

          WWN             Slot Number   Status
10:00:00:05:1e:c1:75:ac        0        Online
10:00:00:05:1e:54:22:75        0        Online
```

```
Device Configuration Details:
---------------------------------

Crypto Device Information:

Container name:          Symm_10G0_R1
Type:                    disk
EE node:                 10:00:00:05:1e:c1:75:ac
EE slot:                 0

Container name:          Symm_10G1_R1
Type:                    disk
EE node:                 10:00:00:05:1e:54:22:75
EE slot:                 0
Operation succeeded.


i2051131:admin> cryptocfg --show -egstatus -stat | more

Encryption Group Details:
---------------------------------

Encryption Group Information
            EG Name:                    R1_131_136
            EG state:                   CLUSTER_STATE_CONVERGED
            Failback mode:              Auto
            Replication mode:           Enabled
            Heartbeat misses:           3
            Heartbeat timeout:          2
            Number of defined nodes:    2
            Group Leader Node Name:     10:00:00:05:1e:c1:75:ac

        Node Name:                      10:00:00:05:1e:c1:75:ac  (current node)
            State:                      DEF_NODE_STATE_DISCOVERED
            Role:                       GroupLeader
            IP Address:                 10.246.51.131
            Certificate:                10.246.51.131_my_cp_cert.pem
            Current Master Key State:   Saved
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
            Alternate Master Key State: Saved
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

    EE Slot:                    0
        SP state:               Online
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
            HA Cluster Membership:      R1_131_136_cluster
```

```
        Node Name:                      10:00:00:05:1e:54:22:75
            State:                      DEF_NODE_STATE_DISCOVERED
            Role:                       MemberNode
            IP Address:                 10.246.51.136
            Certificate:                10.246.51.136_my_cp_cert.pem
            Current Master Key State:   Saved
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
            Alternate Master Key State: Not configured
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

    EE Slot:                    0
        SP state:               Online
            Current Master KeyID:
   1b:91:ea:93:06:33:d5:bb:a6:57:ab:b8:d8:f0:07:69
            Alternate Master KeyID:
   00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
            HA Cluster Membership:      R1_131_136_cluster

Key Vault Details:
----------------------------------

Primary Key Vault Information:
    Key Vault State:    configured
    IP address:         10.32.139.200
    Certificate ID:     sgeliopvm20
    Certificate label:  R1_RKMS
    State:              Connected
    Type:               RKM

  Secondary Key Vault Information:
    Key Vault State:    not configured

Additional Key Vault/Cluster Information:
        Key Vault/CA Certificate Validity:      Yes
        Port for Key Vault Connection:          443
        Time of Day on Key Server:              N/A
        Server SDK Version:                     N/A

Encryption Node (Key Vault Client) Information:
        Node KAC Certificate Validity:          Yes
        Time of Day on the Switch:              N/A
        Client SDK Version:                    RKM-Client 2.1.1 27-September-2007
        Client Username:                        N/A
        Client Usergroup:                       N/A
        Connection Timeout:                     3 seconds
        Response Timeout:                       25 seconds
        Connection Idle Timeout:                N/A

Key Vault configuration and connectivity checks successful, ready for key
   operations.
```

```
HA Cluster Details:
---------------------------------

HA Cluster State:       Configured
Number of HA Clusters:  1

HA cluster name: R1_131_136_cluster - 2 EE entries
Status:          Committed
HAC State:       Converged

          WWN              Slot Number   Status
10:00:00:05:1e:c1:75:ac         0        Online
10:00:00:05:1e:54:22:75         0        Online

Device Configuration Details:
---------------------------------

Crypto Device Information:
Container name:          Symm_10G0_R1
EE node:                 10:00:00:05:1e:c1:75:ac
EE slot:                 0

LUN serial number:       60000970000192603025533030343837
Lun State:               Encryption enabled
Replication Lun State:   Primary
LUN  DEK if Diff EncMode EncFormat EncExistingData  ReKeyState KeyLife
   LunType
0x1  0           1       0         0                0          0              disk

Operation succeeded.
```

Note: A detailed explanation of the rekeying process for SRDF LUNs is included in the *Fabric OS Encryption Administrator's Guide Supporting RSA Key Manager (RKM) Environments, Supporting Fabric OS v6.4.0*, which can be located under the Documents section of MyBrocade located at https://login.brocade.com.

Consider the following when rekeying encrypted SRDF LUNs:

- Auto rekey is disabled for replicated LUNs. Sync between primary LUNs and mirror LUNs should be disabled before starting manual rekey on primary LUNs. If sync is not disabled, the mirror LUN will be disabled for host access. Once the primary LUN rekey completes, sync can be performed between primary and mirror LUN.
- Manual rekey works only on primary LUNs. Mirror LUNs can be converted to primary LUNs by performing a manual rekey with **-include_mirror** option.

- **cryptocfg - - manual_rekey -all -include_mirror** rekeys all the primary and mirror LUNs, not just mirror LUNs and out-of-sync primary LUNs. Only type **cryptocfg - - manual_rekey -all** if you want to rekey only out-of-sync primary LUNs. The **-include_mirror** option is ignored if the command applies only to a primary LUN.

RecoverPoint encryption case study

This case study describes how to deploy Brocade fabric-based encryption in a two-site configuration in which EMC RecoverPoint is used to replicate encrypted data between the two sites. The following information is included:

- "Configuration requirements" on page 293
- "Reference architecture" on page 294
- "Summary of initialization steps" on page 295
- "Rekeying in the RecoverPoint environment" on page 299

Assumptions

For the steps that follow, this case study assumes the following:

- There is a clustered pair of RSA Key Managers (RKMs) at the local site and a clustered pair of RKMs at the remote site.
- The clustered pairs must be configured as part of the same RKM group.

Configuration requirements

The following are configuration requirements for RecoverPoint encryption solutions:

- Encryption RecoverPoint LUNs can be done only when the key vault configured is the RKM.
- Currently, encryption is only supported for RecoverPoint consistency groups utilizing array-based splitters.
- The RecoverPoint appliances must *not* be configured in a CTC. All appliance or storage connections must be cleartext only.
- Replication feature needs to be enabled.

Note: If replication is enabled, firmware downgrades to older FOS releases will be disallowed until the replication feature is disabled. The replication feature cannot be disabled if there are LUNs in the encryption group (EG) configured with the **–newLUN** option.

Reference architecture

The reference architecture for this case study is shown in Figure 38.

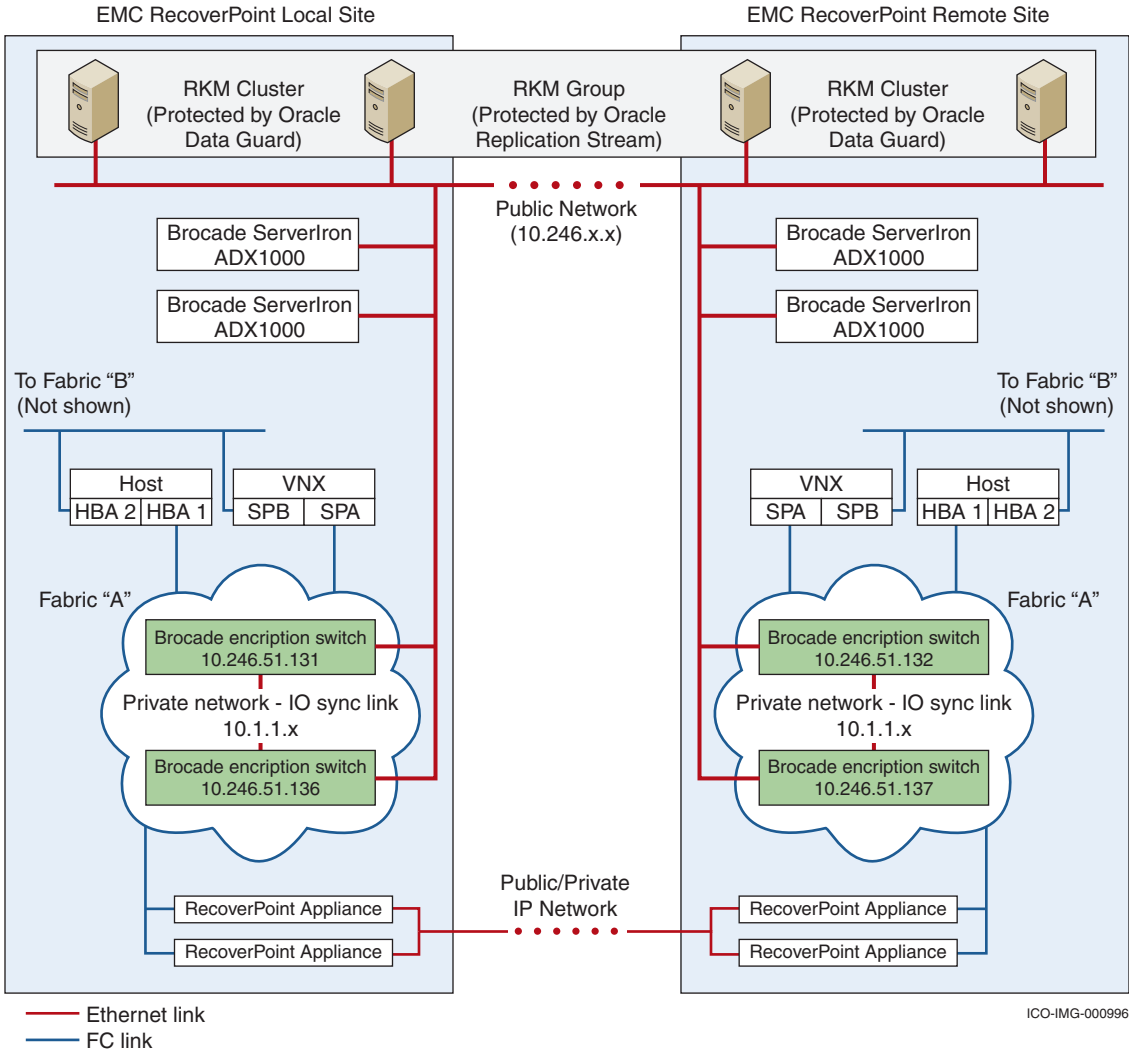


Figure 38 RecoverPoint case study architecture

As shown in Figure 38 on page 294, each of the two sites in the architecture consists of both Fabric A and Fabric B. For example, FabricA at Site 1 (Local) consists of two Brocade Encryption switches. Likewise, Fabric A at Site 2 (Remote) also consists of two Brocade Encryption. Each site will be made up of a single Encryption Group (EG) as shown in the architecture.

For ease of explanation, the configuration steps in the following section will only provide step-by-step instructions for setting up Fabric A at each site. That is, each site's Fabric B setup steps will not be addressed. The setup steps for each Fabric B would be nearly identical to the steps involved for setting up Fabric A.

Summary of initialization steps

The steps for initializiang and configuring encryption in a RecoverPoint environment are the same as in the "SRDF encryption case study" on page 232. Follow the steps listed in "Configuring the Encryption Engines" on page 236 through "Ensuring that both fabrics are set for defzone--noaccess" on page 276.

IMPORTANT

Setting up Brocade fabric-based encryption requires initialization steps, which are performed only once and which must be executed in the specified order indicated.

The most challenging aspect of setting up an encryption solution involves the initial configuration of the Brocade encryption switches, RKM KVs, and the IPLBs. Once communication has been established between all encryption switches in an EG and their associated RKM KVs, the majority of the setup work has been completed.

The final steps of an encryption setup consisting of configuring storage targets (CTCs) and their associated LUNs are straightforward and take far less time.

Assumptions

The following is assumed:

- All of the encryption switches, and RKM KVs, and IPLBs have been powered on.
- All of the Fibre Channel cabling, management interface cabling, and I/O Sync Link cabling between encryption engines have been completed.

After completing the steps listed in the "Configuring the Encryption Engines" on page 236 through "Ensuring that both fabrics are set for defzone--noaccess" on page 276 sections, the following steps should also be completed:

- "Step 1: Creating the CTCs at each site" on page 296
- "Step 2: Adding the source RecoverPoint LUNs to each CTC at Site 1 (Local)" on page 297
- "Step 3: Adding the remote/target RecoverPoint LUNs to each CTC at Site 2 (Remote)" on page 298

Step 1: Creating the CTCs at each site

The CTCs which will be part of the encrypted traffic flow must now be added to the EEs at each site. For the purposes of this reference, only the CTCs which are part of each site's Fabric A will be displayed. When setting up this solution for a customer, care must be taken to ensure that all CTCs from all fabrics participating in the RP Solution are added to the appropriate EEs.

For this RP encryption solution, two CTCs (VNX Storage Ports) will be added to each Site's Fabric A configuration. One of the two CTCs at each site will be owned by one of the two HA cluster members at each site, making each HA cluster member active.

From the Site 1 EG leader node:

```
i2051131:admin> cryptocfg --create -container disk VNX_SPA_R1
10:00:00:05:1e:c1:75:ac 50:06:01:6b:46:e0:02:f9 50:06:01:60:c6:e0:02:f9
Slot Local/
EE Node WWN Number Remote
10:00:00:05:1e:c1:75:ac 0 Local
Operation succeeded.
i2051131:admin> cryptocfg --add -initiator VNX_SPA_R1 10:00:00:00:c9:8f:58:58
20:00:00:00:c9:8f:58:58
Operation succeeded.
```

The initiator is added at the CTC level to allow discovery of the LUNs behind the storage port to which the added initiator has access.

```
i2051131:admin> cryptocfg --create -container disk VNX_SPB_R1
10:00:00:05:1e:54:22:75 50:06:01:63:46:e0:02:f9 50:06:01:60:c6:e0:02:f9
Slot Local/
EE Node WWN Number Remote
10:00:00:05:1e:54:22:75 0 Local
Operation succeeded.
i2051131:admin> cryptocfg --add -initiator VNX_SPB_R1 10:00:00:00:c9:8f:58:58
20:00:00:00:c9:8f:58:58
Operation succeeded.
```

Once the commit operation is complete, all I/O to all LUNs via the configured storage port will be stopped. This is because the commit operation results in Frame Redirection occurring through the fabric. Therefore, any I/O destined to the configured storage ports will now be going through the EEs and because the EEs have yet to be configured with actual LUNs, all I/O will be halted.

```
i2051131:admin> cryptocfg --commit
Operation succeeded.
```

A maximum of 25 transactions per **commit cryptocfg -commit** can be executed.

```
i2051131:admin> cryptocfg --show -container -all -cfg
Encryption group name: R1_131_136
Number of Container(s): 2
Container name: VNX_SPA_R1
Type: disk
EE node: 10:00:00:05:1e:c1:75:ac
EE slot: 0
Target: 50:06:01:6b:46:e0:02:f9 50:06:01:60:c6:e0:02:f9
VT: 20:02:00:05:1e:54:22:40 20:03:00:05:1e:54:22:40
Number of host(s): 1
Configuration status: committed
Host: 10:00:00:00:c9:8f:58:58 20:00:00:00:c9:8f:58:58
VI: 20:04:00:05:1e:54:22:40 20:05:00:05:1e:54:22:40
Number of LUN(s): 0
Container name: VNX_SPB_R1
Type: disk
EE node: 10:00:00:05:1e:54:22:75
EE slot: 0
Target: 50:06:01:63:46:e0:02:f9 50:06:01:60:c6:e0:02:f9
VT: 20:00:00:05:1e:54:22:40 20:01:00:05:1e:54:22:40
Number of host(s): 1
Configuration status: committed
Host: 10:00:00:00:c9:8f:58:58 20:00:00:00:c9:8f:58:58
VI: 20:06:00:05:1e:54:22:40 20:07:00:05:1e:54:22:40
Number of LUN(s): 0
Operation succeeded.
```

Step 2: Adding the source RecoverPoint LUNs to each CTC at Site 1 (Local)

There are two possible LUNs configurations to consider in encryption RecoverPoint deployments:

- Creating new source LUNs which can later be replicated.
- Migrating data from existing source LUNs to LUNs that can be replicated. This solution creates new source LUNs and replicates them to a remote site.

For each scenario, the following rules and notes apply:

- RecoverPoint source and target LUNs must be the same size.
- When changing encryption policies for the source LUN, the same policies must be applied to the target LUN.
- Once the LUN is added to a container, it can not be resized.
- Auto/Key expire rekey is not allowed for RecoverPoint LUNs.

Once the CTCs are created and its configurations committed, use the **cryptocfg –discoverLUN** command for LUN discovery. This command is used on the CTC level and discovers and displays all LUNs behind the storage port for which initiators that are part of the CTC have access to. Refer to "Adding the source SRDF LUNs to each CTC at Site 1" on page 280 for example output.

Step 3: Adding the remote/target RecoverPoint LUNs to each CTC at Site 2 (Remote)

This section describes the proper procedure for establishing the local or remote LUN pair in a RecoverPoint environment.

Note: The remote or target LUNs should be added to the CTC only after the local site LUN's encryption setup has been completed.

Establish the RecoverPoint consistency group (CG) and wait for the initial synchronization to be completed and the CG to be in the active state. You can verify the RecoverPoint pair is in a synchronized state using the RecoverPoint GUI.

Note: During RecoverPoint replication, the remote or target LUNs will not be visible to remote site hosts.

Refer to "Adding the source SRDF LUNs to each CTC at Site 2" on page 284 for examples and sample output.

Rekeying in the RecoverPoint environment

Consider the following when rekeying encrypted RecoverPoint LUNs:

- Auto Rekey is disabled for replicated LUNs.
- Replication between source LUNs and target LUNs should disabled before starting a manual rekey operation on the primary LUN.
- Once the rekey operation is complete, replication should be enabled, which causes a full sweep and replicates the newly rekeyed data to the target volume.

8

Best Practices for EMC Disk Library and Encryption Switches

This chapter discusses the best practices of switch encryption implemented in front of the EMC Disk Library (EDL). The key manager in this solution is RSA Key Manager. The following information is provided in this section:

- Overview 302
- Challenges 304
- Best practices for Cisco SME and Brocade Encryption Switch ... 305
- Turning compression on and off on the EDL 309

Overview

RSA, the security division of EMC, is the premier provider of security solutions for business acceleration, helping the world's leading organizations succeed by solving their most complex and sensitive security challenges. RSA's information-centric approach to security guards the integrity and confidentiality of information throughout its lifecycle, no matter where it moves, who accesses it, or how it is used. RSA offers industry-leading solutions in identity assurance and access control, data loss prevention, encryption and key management, compliance and security information management and fraud protection. More information can be found at www.RSA.com.

RSA Key Manager is an enterprise key management solution that can:

- Increase security by giving granular control of encryption policies.
- Lower the total cost of ownership associated with encryption by automating tasks.
- Manage keys across all layers of the infrastructure stack including:
  - Applications
  - Databases
  - SAN networking
  - Disk storage systems
  - Tape storage systems

The EMC Disk Library delivers industry-leading scalability, performance, and availability, while emulating leading tape solutions. EDL is a simple to deploy to easy to use solution that enables disk-based backup and restore without chaning current operations or backup infrastructure.

Encryption is the process of encoding or converting data in clear text to an encrypted form, called *cipher text*, which cannot be decoded by unauthorized person or software without an encryption key, as shown in Figure 39 on page 303.

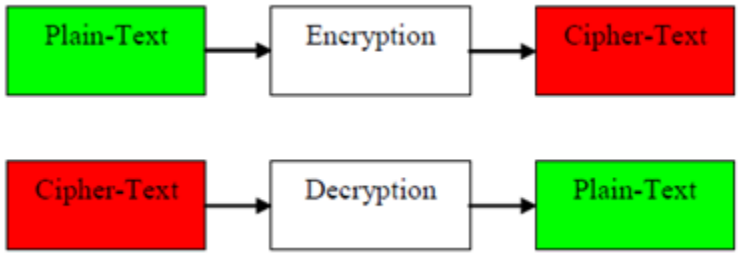


Figure 39 Encryption process

In the storage networking industry, encryption can be implemented anywhere in the network, from the host to the target. Possible solutions include host-based, target-based, or network-based.

The encryption key is managed by an encryption key manager. For tape encryption, each tape or virtual tape gets its own unique key ID. The actual encryption key is stored on the RSA RKM key manager server. That key ID stays with the tape or virtual tape for the life of the tape, or until it is manually changed by the administrator.

Challenges

Consider the following challenges:

- Both the EDL and the encryption switch are capable of compressing the data. Compressing already compressed data will not be effective, or can sometimes be lossy. Furthermore, properly encrypted data cannot be compressed. As a result of compressing encrypted data, backup capacity can extend beyond the original capacity prior to encryption.
- An encryption switch adds a header of significant size to every data block. As a result there is an overhead created by the encryption switch.
- The size of every data block written using some backup applications is random and uncompressible.
- The EDL is a block device internally and stores the data in terms of blocks, although it represents itself as a tape, which is a serial device. Therefore, the effective tape size is not the same as declared.
- EDL tape export does not support tape spanning. Therefore, the overhead of the encryption switch may require the user to adjust the capacity of the cartridge.

Best practices for Cisco SME and Brocade Encryption Switch

Based on the implementation of encryption by the Cisco Storage Media Encryption (SME) and the Brocade Encryption Switch , EMC recommends the following best practices.

Cisco SME

Cisco Storage Media Encryption (SME) protects data at rest on heterogeneous tape drives and virtual tape libraries (VTLs) in a SAN environment using secure Advanced Encryption Standard (AES) 256 algorithms.

The following hardware includes encryption units suitable for Cisco SME

- Cisco MDS 9222i Multiservice Modular Switch (MMS)
- Cisco MDS 9000 18/4-Port Multiservice Module (MSM)
- Cisco MDS 9000 16-Port Storage Services Node (SSN)

Cisco SME encrypts and compresses the data received and adds a header of 64K for each block of data received before forwarding it to target. In case of uncompressible data, SME will add a 64K header to the uncompressed data blocks. An example is shown in Figure 40 on page 306.

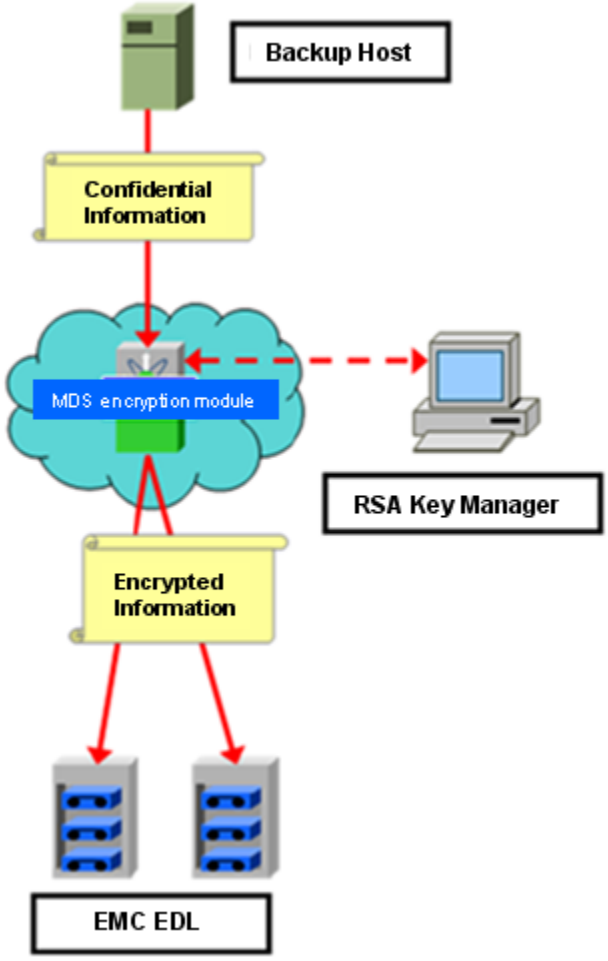


Figure 40 **Cisco SME encryption example**

For details of the configuration of Cisco SME, refer to www.cisco.com.

The following are the best practices to be followed when using Cisco MDS encryption module in front of the EDL through the FC port.

- EDL compression must be turned off.
- If export to tape is used, the capacity of the virtual cartridge should be smaller than the physical tape it is exported to.

Be aware that:

- A performance decrease may be seen with encryption.
- Due to the random block size of the encrypted data, the effective tape size of the virtual cartridge can be smaller than expected.

Brocade Encryption Switch

The Brocade Encryption Switch is a 32-port 8 Gb/s Fibre Channel switch with encryption, decryption, and compression capabilities. It encrypts the data using Advanced Encryption Standard (AES) 256-bit algorithms. It provides fabric-based encryption to protect digital assets in enterprise data center environments. An example of the Brocade Encryption Switch deployment is shown in Figure 41.

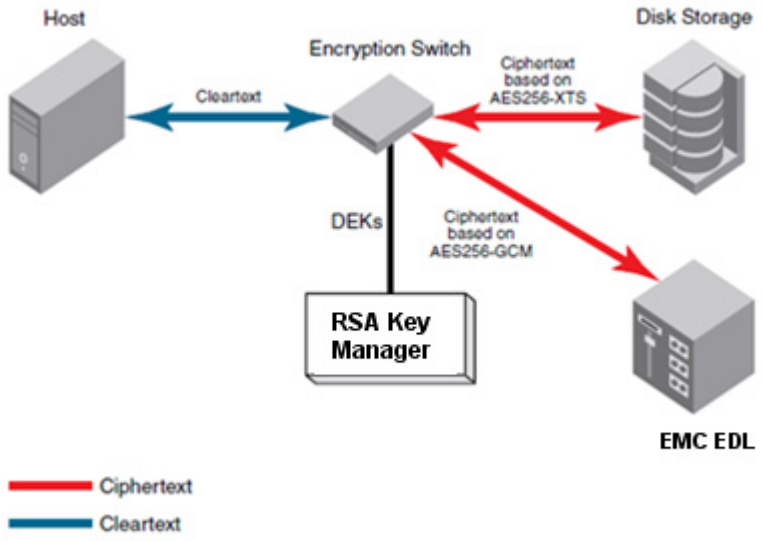


Figure 41 **Brocade Encryption Switch encryption example**

The FS8-18 blade provides the same features and functionalities as the Brocade Encryption Switch. FS8-18 can be installed in DCX and DCX-4S.

The Brocade Encryption Switch encrypts and compresses the data received and pads with zeros to make it to the original size, adds a header of 1K to each of the block and transfers to the target.

When the EDL compresses the encrypted data by the Brocade Encryption Switch, it will remove all the zeros.

For details on the configuration of the Brocade Encryption Switch, refer to the www.brocade.com.

The following are the best practices to be followed when using the Brocade Encryption Switch in front of EDL FC port.

- EMC supports the Brocade Encryption Switch in front of the EDL with the following limitations:
  - The maximum number of VTL LUNs per CTC is eight, including medium changers and the tape drives.
- EDL compression must be turned on.
- If export to tape is used, to avoid physical tape spanning, the size of the virtual media needs to be smaller than the physical tape it is exported to.

Be aware that:

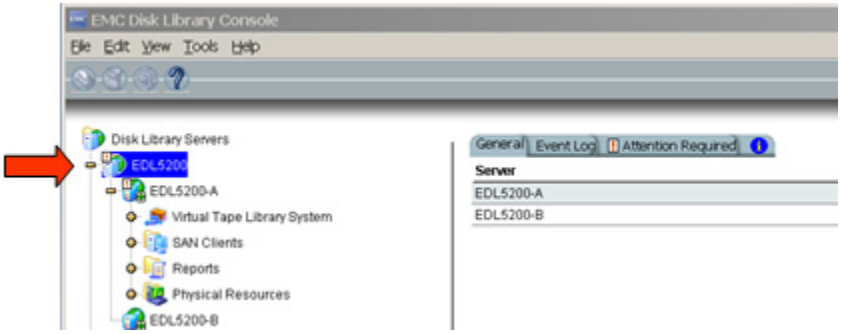
- A performance decrease may be seen with encryption.
- If export to tape is used and the application is using the random block size writes, the effective tape size of the virtual volume can be smaller than expected due to the additional header added to the encrypted data blocks.

Turning compression on and off on the EDL

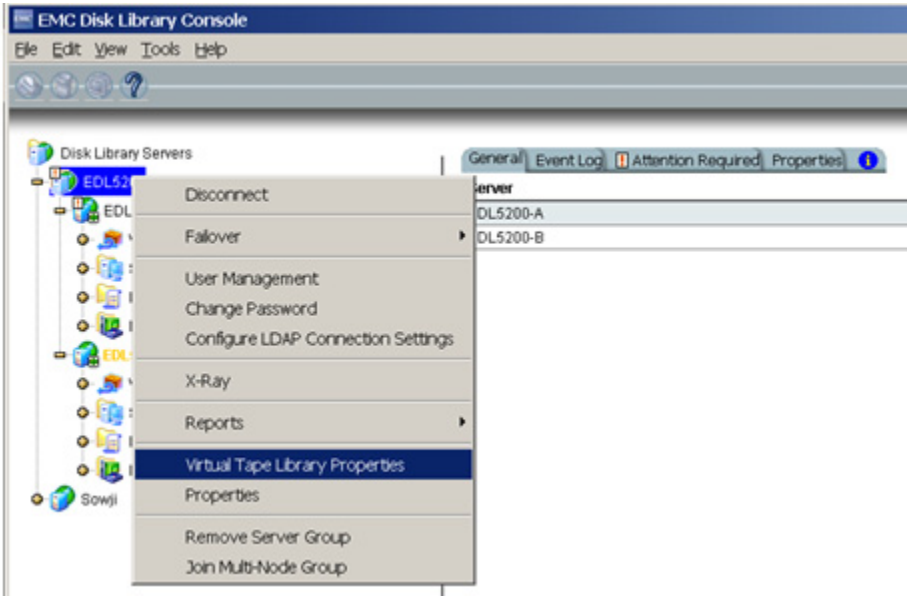
To turn compression on and off on the EDL, complete the following steps:

1. Log in to the EDL console and right-click on the EDL server.

   If there are two nodes, right-click on the main EDL server (not EDL -A or B).



2. Select **Virtual Tape Library Properties**.



A **Change Virtual Tape Library properties** dialog box displays.

3. By *default*, compression is *disabled* on the EDL.

   To *enable* compression mode for the VTL, select the E**nable Virtual Tape Library compression mode** checkbox and click **OK**.